Automaton Structures
–
Decision Problems and Structure Theory

Von der Fakultät für Informatik, Elektrotechnik und Informationstechnik
der Universität Stuttgart zur Erlangung der Würde eines
Doktors der Naturwissenschaften (Dr. rer. nat.)
genehmigte Abhandlung

Vorgelegt von
Jan Philipp Wächter
aus Schwaikheim

Hauptberichter:	Volker Diekert
Mitberichter:	Alan J. Cain und Manfred Kufleitner
Tag der mündlichen Prüfung:	18. Dezember 2020

Institut für Formale Methoden der Informatik
der Universität Stuttgart
2020

Contents

-1 Introduction **1**

0 Preliminaries **5**
0.1 Fundamentals and Notation . . . 5
0.2 Automata, Constructions and Structures . . . 8
0.2.1 Automata . . . 8
0.2.2 Automaton Constructions . . . 11
0.2.3 Automata and Algebraic Structures . . . 13
0.3 Automaton Actions . . . 16
0.3.1 The Dual Action . . . 16
0.3.2 Stabilizers, Orbits and Orbital Graphs . . . 17

1 Structure Theory **21**
1.1 Partial and Complete Automata . . . 21
1.1.1 Re-Using and Adjoining Zeros . . . 22
1.1.2 Removing Zeros . . . 25
1.1.3 A Problem Due to Cain . . . 28
1.2 Non-Automaton Semigroups . . . 29
1.2.1 Semidirect Products of the Monogenic Free Semigroup . . . 30
1.3 Inverse Automaton Semigroups . . . 33
1.3.1 The Monogenic Free Inverse Monoid . . . 34
1.3.2 Inverse Automaton Semigroups and Automaton-Inverse Semigroups 37
1.4 Orbits of Automaton Semigroups . . . 45
1.4.1 Infinite Orbits . . . 46
1.4.2 Orbits of Periodic and Non-Periodic Words . . . 54

2 Decision Problems **59**
2.1 Word Problem . . . 59
2.1.1 A Straight-Forward "Guess and Check" Algorithm . . . 61
2.1.2 Simulating a Turing Machine . . . 65
2.1.3 Compressed Word Problem . . . 78
2.2 Positive Relations . . . 85
2.2.1 A Construction Due to Šunić and Ventura . . . 86
2.2.2 The Reduction . . . 90
2.2.3 Interlude: the Freeness Problem . . . 93
2.3 Finiteness Problem . . . 97
2.3.1 An Undecidability Result Due to Lukkarila . . . 98

2.3.2 The Reduction 101
2.3.3 The Case of Invertible Automaton Structures 103
2.4 Expandability 108
2.4.1 Expandability is Decidable 110
2.4.2 Expandability in Groups 114
2.4.3 Groups of Bounded Activity 117

Bibliography **125**

Abstract

This thesis is devoted to the class of automaton groups and semigroups, which has gained a reputation of containing groups and semigroups with special algebraic properties that are hard to find elsewhere. Both automaton groups and semigroups are studied from a structural and an algorithmic perspective. We motivate the use of partial automata as generating objects for algebraic structures and compare them to their complete counterparts. Additionally, we give further examples of semigroups that cannot be generated by finite automata and show that every inverse automaton semigroup is generated by a partial, invertible automaton. Moreover, we study the finite and infinite orbits of ω-words under the action induced by an automaton. Here, our main result is that every infinite automaton semigroup admits an ω-word with an infinite orbit.

We apply these structural results algorithmically and show that the word problem for automaton groups and semigroups is PSPACE-complete. Furthermore, we investigate a decision problem related to the freeness of automaton groups and semigroups: we show that it is undecidable whether a given automaton admits a non-trivial state sequences that acts trivially and we use this problem for further reductions. Afterwards, we strengthen Gillibert's result on the undecidability of the finiteness problem for automaton semigroups and give a partial solution for the group case of the same problem. Finally, we consider algorithmic questions about increasing the orbits of finite words and apply these results to show that, among others, the finiteness problem for (subgroups of) automaton groups of bounded activity is decidable.

Zusammenfassung

Diese Arbeit widmet sich der Klasse der Automatengruppen und -halbgruppen, die bekannt dafür ist, Gruppen und Halbgruppen mit speziellen Eigenschaften zu enthalten, die sich anderswo nur schwer finden lassen. Sowohl Automatengruppen als auch -halbgruppen werden aus einem strukturellen und einem algorithmischen Blickwinkel untersucht. Wir motivieren die Verwendung partieller Automaten als erzeugende Objekte algebraischer Strukturen und vergleichen sie mit ihren vollständigen Gegenstücken. Außerdem geben wir weitere Beispiele von Halbgruppen an, die nicht durch endliche Automaten erzeugt werden können, und zeigen, dass alle inversen Automatenhalbgruppen von einem partiellen, invertierbaren Automaten erzeugt werden. Zudem untersuchen wir die endlichen und unendlichen Bahnen von ω-Wörtern unter der durch einen Automaten induzierten Wirkung. Hierbei ist unser Hauptresultat, dass jede unendliche Automatenhalbgruppe ein Wort mit unendlicher Bahn liefert.

Wir wenden diese strukturellen Ergebnisse algorithmisch an und zeigen, dass das Wortproblem für Automatengruppen und -halbgruppen PSPACE-vollständig ist. Darüber hinaus untersuchen wir ein Entscheidungsproblem in Verbindung mit der Freiheit von Automatengruppen und -halbgruppen: Wir zeigen, dass es unentscheidbar ist, ob ein gegebener Automat eine nichttriviale Zustandsfolge besitzt, die trivial operiert, und verwenden dieses Problem für weitere Reduktionen. Anschließend erweitern wir das Ergebnis von Gillibert über die Unentscheidbarkeit des Endlichkeitsproblems für Automatenhalbgruppen und geben eine partielle Lösung für dasselbe Problem im Gruppenfall an. Schließlich betrachten wir algorithmische Fragestellungen hinsichtlich der Vergrößerung der Bahnen endlicher Wörter und wenden diese Ergebnisse an, um zu zeigen, dass unter anderem das Endlichkeitsproblem für (Untergruppen von) Automatengruppen beschränkter Aktivität entscheidbar ist.

-1 Introduction

The intriguing class of automaton groups is a rich source for groups with interesting and sometimes peculiar properties. In contrast to more classical forms of presentation, the groups in this class are defined using automata, which, in this context, refer to what might be more precisely described as finite-state, letter-to-letter transducers. Every such automaton defines an action mapping input to output words, which is then used to generate the group. Due to their nature based on finite-state automata, one might be tempted to think that only structurally simple or even only finite groups can be generated in this way. However, quite on the contrary, even a seemingly simple automaton can generate a group with surprisingly complex behavior. The best known example here is probably the Grigorchuk group. It became famous as the historically first example of a group with subexponential but superpolynomial growth, answering a question by Milnor [Car+68, Problem 5603] on the existence of such groups. The growth of a group refers to the number of elements contained within balls of growing size around the neutral element in the Cayley graph of the group.[1] Additionally, while the Grigorchuk group itself is infinite, every group element has finite order. Thus, it also constitutes an answer with a quite simple presentation to the Burnside problem [Bur02]. The tension between the simplicity of the presentation and the complexity of the behavior also extends to research questions in the area. Even though questions may have simple formulations, answers are often notoriously hard to obtain,[2] even in cases for which the eventual solution may turn out to be rather simple.[3] This leads to the phenomenon that even partial solutions can have significant value.[4]

The idea of using automata to generate groups naturally extends to inverse and non-inverse semigroups. This leads to the term "automaton structures" used in this work to cover automaton groups, inverse automaton semigroups and (general) automaton semigroups. Although the research initially was – and to a certain degree still is – mostly focused on groups, automaton semigroups seem to attract more and more interest lately.[5] One reason for this increase might be that automaton semigroups often come up even if one is mostly interested in automaton groups. For example, this occurs by considering the dual automaton, in which states and letters changes roles. In general, the dual of an automaton generating a group will not generate a group itself; however, the dual can be used to study the original automaton and its generated group.[6] Another reason might be found in the overall landscape of the research activity in the area. In rough terms, the research on automaton structures can be divided into three branches. First (also in a chronological sense), there is the research on individual interesting automaton groups such as the already mentioned Grigorchuk group. The observation that many of these groups can naturally be presented using automata lead to the second research branch focusing on the overall class of automaton groups and on relations between properties of

[1] See [GP08] for an accessible introduction to this topic with an emphasis on the Grigorchuk group.

[2] See for example the problem list in [GNS00]

[3] One could argue that the problem in Section 1.4 is an example for this.

[4] See for example [Kli13; Gil14; Pic19]

[5] See for example [Cai09; BC15; BC17; Kli16; Bar+18; Pic19] to only mention a few.

[6] We will make heavy use of this connection for example in Section 1.4 but also in Section 2.3.

the automaton and properties of the generated group or semigroup. At this point, the third research branch comes into play naturally: it studies algorithms to answer questions on automaton structures. It is this algorithmic study of automaton structures that might explain some of the interest in automaton semigroups. Obviously, the automaton usually plays a central role in algorithms for automaton groups and many times it is more important than the actual group structure. Therefore, algorithms for automaton groups can often be generalized to semigroups. On the other hand, many decision problems over automaton groups are either proven or generally suspected to be undecidable. Since, compared to groups, it is usually easier to encode computations in semigroups, this leads to the situation that undecidability results for automaton semigroups exist while the corresponding problems for groups are still open or could only be solved later. An important example here is the finiteness problem, which asks whether a given automaton generates a finite or an infinite group or semigroup, respectively. It has been shown to be undecidable for automaton semigroups by Gillibert [Gil14] but its decidability for groups is still an important open problem [GNS00, 7.2 b)]. Another example is the order problem, where the question is whether a given group element has finite or infinite order. A similar problem for automaton semigroups had already been shown to be undecidable [Gil14, Corollary 3.14] but the undecidability in the group case was only obtained later [Gil18] (also in the case of so-called contracting automaton groups [BM20]). Similarly, it could be shown that the word problem, which, in this case, asks whether two given state sequences represent the same semigroup element, is PSpace-complete for automaton semigroups – in fact, even for inverse automaton semigroups – [1] but the result could only later be lifted to automaton groups [5]. On the other hand, the other two of Dehn's fundamental problems in algorithmic group theory [Deh11], the conjugacy and isomorphism problem, have been settled directly for the group case [ŠV12].

The aim of this work is to contribute to the latter two of the above mentioned research branches: the structural and the algorithmic study of automaton structures. As the theory of automaton structures lies at the meeting point of algebra and automaton theory, it is not surprising that we will combine various tools from different areas of theoretical computer science and mathematics such as complexity and computability, automaton theory, Wang tilings, graph theory and, of course, a bit of semigroup and group theory.

In Chapter 1, we present the structural results, where we put an emphasis on partial automata. This emphasis is motivated by the fact that partial automata have turned out to be useful for certain algorithmic problems[7] and allow for a natural presentation of inverse semigroups using automata. We discuss this and compare the classes of semigroups generated by partial and by complete automata in Section 1.1, where we show some closure properties for the partial case; the corresponding closure properties for the complete case pose an open problem by Cain [Cai09, Open problem 5.3] and we establish that this open problem has a positive answer if and only if both classes coincide. Then, in Section 1.2, we show that (certain subsemigroups of) semidirect products of the monogenic free semigroup are not automaton semigroups, which significantly increases the so-far very limited set of examples for such semigroups. Next, we look at invertible yet partial automata in Section 1.3 and the inverse semigroups they generate. We present the monogenic free inverse monoid as such an automaton semigroup and then show that

[7] namely the word problem (see Section 2.1) and the finiteness problem (see Section 2.3)

every automaton semigroup that happens to be an inverse semigroup can be generated by a partial invertible automaton. For the latter result, we generalize the classical Preston-Vagner theorem (see e. g. [How95, Theorem 5.1.7, p. 150] or [Pet84, p. 168]). Finally, we have a look at the orbits of automaton semigroups in Section 1.4. Our main result here is that every infinite automaton semigroup (and, thus, every infinite automaton group) admits an ω-word with an infinite orbit, which solves an open problem communicated by Ievgen Bondarenko (see also [3, Open problem 4.3]). While this is an important result on its own, the techniques used and developed in the course of its proof are also quite versatile and we apply them to some other problems.

Chapter 2 collects some algorithmic results for automaton structures (where we also make use of the previously shown structural results). We start with a study of the word problem for automaton groups and semigroups in Section 2.1 and show that the uniform and non-uniform versions of this problem are PSpace-complete, which proves a conjecture by Steinberg [Ste15, Question 5] (see also [AIM07, section 2, 6.]) and is related to a problem by Cain [Cai09, Open problem 3.6]. The basis of this proof is a general construction to simulate Turing machines in automaton groups, which we also apply to the so-called compressed word problem. Then, in Section 2.2, we consider a problem related to the freeness of automaton semigroups and groups: we show that it is impossible to decide whether a given automaton admits a non-trivial state sequence which acts trivially. We motivate this problem by reducing it to some other natural problems and conclude the section with a discussion of why the used constructions cannot be applied to the freeness problem for automaton semigroups and groups [GNS00, 7.2 b)], which asks whether a given automaton generates a free automaton structure. In Section 2.3, we discuss the finiteness problem for automaton structures. We strengthen Gillibert's result on the general undecidability in the semigroup case [Gil14] to bi-reversible yet partial automata. We briefly discuss the problem of extending the presented construction to inverse semigroups and give a partial result to the finiteness problem for automaton groups. The last section, Section 2.4, is devoted to the notion of expandable words. These are words whose orbit size can be increased by appending a suffix. We see that the expandability of a given word can be decided, even for semigroups, and give a more efficient algorithm in the group case. In both cases, we also obtain upper bounds on the length of the orbit increasing suffix. Finally, we apply these results to the important class of automaton groups of bounded activity. We obtain that the orbit sizes for such groups can be described using a finite weighted automaton, which allows us to decide many problems that are open or undecidable in the general case. Most notably, we can decide the finiteness problem for this class of automaton groups and the problem asking whether a given automaton acts spherically transitive, which is an open problem in general [GNS00, 7.2 e) and f)]. In fact, we can even solve the finiteness problem for finitely generated subgroups in this setting answering [2, Question 3.7] positively.

0 Preliminaries

0.1 Fundamentals and Notation

Words and Alphabets with Involution. An *alphabet* is a non-empty finite set A. Its elements $a \in A$ are called *letters*. A *finite word* (over A) is a sequence $w = a_1 \dots a_\ell$ with $a_1, \dots, a_\ell \in A$; its *length* is $|w| = \ell$. The set of finite words over A is denoted by A^*. It includes, in particular, the empty word ε and we use $A^+ = A^* \setminus \{\varepsilon\}$ if we want to exclude it. A right infinite sequence $\alpha = a_1 a_2 \dots$ with $a_1, a_2, \dots \in A$ is an *ω-word* (over A). The set of ω-words over A is denoted by A^ω. Finally, the generic term *word* can refer to both finite words and ω-words, and the set of words over A is $A^\infty = A^* \cup A^\omega$. A subset $L \subseteq A^\infty$ is called a *language* (over A).

While we can concatenate a finite word $u \in A^*$ with any word $v \in A^\infty$ to obtain a new word uv, we can only append the empty word to an ω-word $\alpha \in A^\omega$: $\alpha\varepsilon = \alpha$.

For a finite word $w = a_1 \dots a_\ell$ with $a_1, \dots, a_\ell \in A$, we use $\partial w = a_\ell \dots a_1$ to denote its *reverse*.[8] Taking the reverse of an ω-word $\alpha = a_1 a_2 \dots$ with $a_1, a_2, \dots \in A$ leads to a left infinite sequence $\dots a_2 a_1 = \partial\alpha$. We use $A^{-\omega}$ to denote the set of left infinite sequences over A; however, we will not use the term "word" to refer to such objects in order to make a clearer distinction.

[8] The reason for this notation will become apparent later.

To denote subsets and elements of A^∞, we use some natural and common notation. For a language $L \subseteq A^*$ of finite words, we write L^+ for $L^+ = \{w_1 \dots w_i \mid i > 0, w_1, \dots, w_i \in L\}$ and L^* for $L^* = L^+ \cup \{\varepsilon\}$. In addition, we apply operators of words also to languages and write, for example, ∂L for the reverse $\partial L = \{\partial w \mid w \in L\}$ of the language $L \subseteq A^\infty$ and KL for the product $KL = \{uv \mid u \in K, v \in L\}$ of two languages $K \subseteq A^*$ and $L \subseteq A^\infty$. For singleton languages $\{w\}$, we sometimes simply write w and, accordingly, we use w^* for the set $\{w^i \mid i \geq 0\}$ when w is a finite word. We also write w^ω for the ω-word arising from concatenating $w \in A^+$ infinitely many times with itself. An ω-word arising in this way, i. e. one of the form w^ω, is called *periodic* and one of the form uv^ω for $u, v \in A^*$ is called *ultimately periodic*. Similarly, we write $w^{-\omega}$ for the left infinite sequence $\dots ww \in A^{-\omega}$ if w is a finite word from A^+. Furthermore, for a language L of finite words, we write L^n for the set $\{w_1 \dots w_n \mid w_1, \dots, w_n \in L\}$, $L^{\leq n}$ for the set $\{w_1 \dots w_i \mid i \leq n, w_1, \dots, w_i \in L\}$ and $L^{<n}$ for $\{w_1 \dots w_i \mid i < n, w_1, \dots, w_i \in L\}$.

A finite word $u \in A^*$ is a *prefix* of $v \in A^\infty$ if there is some $x \in A^\infty$ with $v = ux$ and $\operatorname{Pre} v$ is the language of all finite prefixes of $v \in A^\infty$. Symmetrically, $u \in A^*$ is a *suffix* of $v \in A^* \cup A^{-\omega}$ if there is some $x \in A^* \cup A^{-\omega}$ with $xu = v$ and $\operatorname{Suf} v$ is the language of all finite suffixes of $v \in A^* \cup A^{-\omega}$. For an ω-word $\alpha \in A^\omega$, we have, in particular, $\operatorname{Suf} \partial\alpha = \partial \operatorname{Pre} \alpha$. A language $L \subseteq A^\infty$ is *prefix-closed* if we have $\operatorname{Pre} w \subseteq L$ for all $w \in L$ and it is *suffix-closed* if we have $\operatorname{Suf} w \subseteq L$ for all $w \in L$.

For every alphabet A, we define a disjoint copy $\overline{A} = \{\bar{a} \mid a \in A\}$ of A and extend the notation $\bar{a}$ into an involution by setting $\bar{\bar{a}} = a$. We write $\widetilde{A}$ for $\widetilde{A} = A \cup \overline{A}$ and, finally, extend the involution to finite words over $\widetilde{A}$ by defining $\overline{a_1 \dots a_n} = \overline{a_n} \dots \overline{a_1}$ for $a_1, \dots, a_n \in \widetilde{A}$. For $L \subseteq \widetilde{A}^*$, we let $\overline{L} = \{\overline{w} \mid w \in L\}$ and $\widetilde{L} = L \cup \overline{L}$ analogously.

Semigroups, Inverse Semigroups, Monoids and Groups. A *semigroup* S is a set equipped with a usually implicit associative binary operation (written in infix notation). An element $z \in S$ is called a *left zero* if we have $zs = z$ for all $s \in S$. Symmetrically, it is a *right zero* if we have $sz = z$ for all $s \in S$ and it is a *zero* if it is both a left and a right zero. A zero in a semigroup is unique if it exists and we can adjoin a new (disjoint) zero $\mathbb{0}$ to any semigroup S by letting $\mathbb{0}s = s\mathbb{0} = \mathbb{0} = \mathbb{0}\mathbb{0}$ for all $s \in S$; this way, we obtain the semigroup $S^{\mathbb{0}}$.

An element $s \in S$ may have a *semigroup inverse*; this is an element $t \in S$ with $sts = s$ and $tst = t$. In general, a semigroup element can have multiple semigroup inverses. However, if every element of a semigroup S has a unique semigroup inverse, then we say that S is an *inverse semigroup*. In this case, we use the notation $\bar{s}$ to denote the semigroup inverse of $s \in S$. Idempotents commute in an inverse semigroup; in fact, a semigroup is inverse if and only if every element has a semigroup inverse and idempotents commute (see e. g. [How95, Theorem 5.1.1, p. 145]). Because $s\bar{s}$ and $\bar{s}s$ are idempotent for all elements s of an inverse semigroup S, we have $\overline{st} = \bar{t}\bar{s}$ for all $s, t \in S$.

A *neutral* element of a semigroup S is an element $e \in S$ with $es = se = s$ for all $s \in S$. If a semigroup S contains a neutral element, it is unique and we denote it by $\mathbb{1}_S$; furthermore, S is called a *monoid* in this case. If the semigroup is clear from the context, we simply write $\mathbb{1}$ instead of $\mathbb{1}_S$. To a semigroup S, we can adjoin a new (disjoint) neutral element $\mathbb{1}$ to obtain the monoid $S^{\mathbb{1}}$ by letting $\mathbb{1}\mathbb{1} = \mathbb{1}$ and $\mathbb{1}s = s = s\mathbb{1}$ for all $s \in S$.

In the case of a monoid M, it makes sense to consider a different notion of inverses: an element $s \in M$ is *(group) inverse* to an element $t \in M$ if we have $st = ts = \mathbb{1}$. A (group) inverse is always unique and every (group) inverse is a semigroup inverse but the converse does not hold. To emphasize the difference between the two concepts of inverses, we sometimes use s^{-1} to denote the (group) inverse of s. If every element of a monoid M has a (group) inverse, then M is a *group*. However, an *inverse monoid* is a monoid that is inverse in the semigroup sense. In this work, we use *algebraic structure* as an umbrella term for (inverse and non-inverse) semigroups, monoids and groups.

An element s of a semigroup S has *torsion* if there are $i, j > 0$ with $i \neq j$ but $s^i = s^j$. The corresponding notion for groups is the *order* of a group element g. It is the smallest number $i > 0$ such that $g^i = \mathbb{1}$; if not such number exists, the order of g is infinite. It is easy to see that an element of a group has torsion if and only if it has finite order. A semigroup is a *torsion* semigroup (or group) if all its elements have torsion. If it does not contain an element of torsion, it is called *torsion-free*.

The set of non-empty finite words Q^+ over some (possibly infinite) set Q forms the *free semigroup* over Q, whose operation is the concatenation of finite words.[9] If we also add the empty word, we obtain the *free monoid* over Q. We obtain the *free inverse semigroup* and the *free inverse monoid* (over Q) from $\widetilde{Q}^*$ and $\widetilde{Q}^+$ if we additionally let $w\overline{w}w = w$ for all $w \in \widetilde{Q}^*$. To obtain the *free group* $\mathscr{F}(Q)$ (over Q) from $\widetilde{Q}^*$, it suffices to let $a\bar{a} = \bar{a}a = \mathbb{1}$ for all $a \in Q$.

[9] In this context only, we also allow "infinite alphabets". It is clear that the definitions for words and languages from above can be extended accordingly.

A semigroup S is *generated* by $T \subseteq S$ if every element of s can be written as a product of elements of T. In the case of a monoid M, we also allow the empty product for the neutral element. An inverse semigroup, inverse monoid or group is generated by some T if it is generated by $\widetilde{T}$ as a semigroup or a monoid, respectively. If an algebraic structure S is generated by some set $T \subseteq S$, there is a natural surjective semigroup homomorphism π from T^+, T^*, $\widetilde{T}^+$ or $\widetilde{T}^*$, respectively, onto S. We write $u = v$ *in* S for two finite words u, v (over T or $\widetilde{T}$) if we have $\pi(u) = \pi(v)$; if we have $\pi(u) \neq \pi(v)$, we write $u \neq v$ in S, accordingly. Finally, an algebraic structure is *finitely generated* if it is generated by some finite set and it is *monogenic* if it is generated[10] by a singleton set $\{s\}$.

[10] For structures with inverses, this means, of course, that $\bar{s}$ is also a generator in the semigroup sense.

Numbers, Sets and Functions. For numbers, we will use common notation. For example, we use $\mathbb{Z}$ for the set of integers and $\mathbb{N}$ for the set of natural numbers, which contains zero.

Concerning the notation for sets, we use $A \uplus B$ to denote the disjoint union of two sets A and B. To indicate that f is a partial function f from A to B, we write $f : A \rightharpoonup B$ and, to indicate that f is total, we write $f : A \to B$. The *composition* of $f : A \rightharpoonup B$ and $g : B \rightharpoonup C$ is $g \circ f : A \rightharpoonup C$ with $(g \circ f)(a) = g(f(a))$. Here, we have used the convention that $f(a)$ and every term involving $f(a)$ is undefined if f is undefined on $a \in A$. This means, for example, that $g(f(a))$ can be undefined even if g is a total function, which is in line with the normal rules of partial function composition. The *domain* of a partial function $f : A \rightharpoonup B$ is $\operatorname{dom} f = \{a \mid f(a) \text{ defined}\}$; its *image* is $\operatorname{im} f = \{f(a) \mid f(a) \text{ defined}\}$. The partial functions $A \rightharpoonup A$ form the semigroup $\mathscr{P}(A)$ with composition as its multiplication.

A partial function $f : A \rightharpoonup B$ is *one-to-one* if $f(a_1) = f(a_2)$ implies $a_1 = a_2$ for all $a_1, a_2 \in \operatorname{dom} f$. We use the term *injective* exclusively for total one-to-one functions. A partial function $\overline{f} : B \rightharpoonup A$ is *semigroup inverse* to $f : A \rightharpoonup B$ if $\operatorname{im} \overline{f} = \operatorname{dom} f$, $\operatorname{dom} \overline{f} = \operatorname{im} f$ and $f(\overline{f}(f(a))) = f(a)$ for all $a \in \operatorname{dom} f$ as well as $\overline{f}(f(\overline{f}(b))) = \overline{f}(b)$ for all $b \in \operatorname{dom} \overline{f}$. The set of one-to-one partial functions $A \rightharpoonup A$ forms the so-called *symmetric inverse semigroup* $\mathscr{I}(A)$. It is an inverse subsemigroup of $\mathscr{P}(A)$.

Actions. A *partial left action* of a semigroup S on a set X is a homomorphism $\alpha : S \to \mathscr{P}(X), s \mapsto \alpha_s$ and α_s is (simply) called *the action of* s in this case. The partial left action α is *faithful* if α is injective and it is a *total* partial left action (on X), or simply a *left action* (on X), if all α_s for $s \in S$ are total functions $X \to X$. In addition to (partial) left actions, there are also (partial) right actions: a *partial right action* is an anti-homomorphism $\alpha : S \to \mathscr{P}(X), s \mapsto \alpha_s$, i. e. we require $\alpha(st) = \alpha(t) \circ \alpha(s)$. In the same way as with (partial) left actions, the partial right action is faithful if α is injective, it is total (on X) if all α_s are total functions $X \to X$ and a total partial right action (on X) is a *right action* (on X). Often, we will use infix notation for actions; i. e. we will for example write $s \circ x$ for $\alpha_s(x)$ in the case of a (partial) left action or $x \cdot s$ for $\alpha_s(x)$ in the case of a (partial) right action. Throughout this text, we will also encounter special actions that respect additional structure in S or in X, for example actions of groups (where α is a group homomorphism to the set of permutations over X) or actions of semigroups on semigroups (where α is a semigroup homomorphism to the endomorphism monoid of a semigroup).

Computability and Complexity Theory. We will need some basic notions from computability and complexity theory, which the reader is assumed to be familiar with.[11] Required knowledge is a basic understanding of deterministic and nondeterministic Turing machines and algorithms as well as the concept of decidable and undecidable decision problems. To show the undecidability of a problem, we usually use computable (many-one) reductions or co-reductions (which are reductions to the complement). Configurations of (single tape) Turing machines are written as words of the form $\Delta^* P \Delta^+$ where Δ is the tape alphabet and P is the state set. Any sequence of such configurations forms a computation, which may be valid or invalid.

[11] Missing details can be found in any introductory textbook on the topic (such as [Pap94] but also [HU79]).

By $\text{NSPACE}(s)$, we denote the class of problems that can be solved by a nondeterministic Turing machine with space bound s. Furthermore, we denote the class of problems solvable by deterministic Turing machines in logarithmic space by LogSpace. For the classes PSpace and ExpSpace of problems solvable in polynomial and exponential space, respectively, it does not matter whether deterministic or nondeterministic Turing machines are considered (by Savitch's theorem, see e. g. [Pap94, Theorem 7.5]). The only complexity class with a time bound that we will encounter[12] is NP, the class of problems solvable by nondeterministic Turing machines in polynomial time. Whenever we talk about a hard or complete problem for a complexity class, we define this with respect to (many-one) LogSpace-reductions.

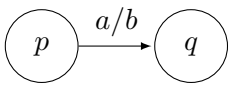
[12] ... and we will only encounter it briefly.

Finally, we make lax use of Landau $\mathcal{O}$-notation (even in multiple variables) but do not go into the formal details.

0.2 Automata, Constructions and Structures

0.2.1 Automata

Automata. The most central notion in this work is that of an automaton. An *automaton* is a triple $\mathcal{T} = (Q, \Sigma, \delta)$ where Q and Σ are alphabets and δ is a relation $\delta \subseteq Q \times \Sigma \times \Sigma \times Q$. To distinguish Q and Σ, we call Q the *state set* of $\mathcal{T}$ and Σ its *(input and output) alphabet*. Accordingly, the elements of Q are called *states* and only the elements of Σ are called *letters* in this context. To distinguish the elements in Q^∞ from those in Σ^∞, we will usually refer to the former as *state sequences* and only to the latter as *words*. Finally, the elements of δ are called *transitions* and, in this context, we use the more graphical notation $p \xrightarrow{a/b} q$ for the tuple $(p, a, b, q) \in Q \times \Sigma \times \Sigma \times Q$. Additionally, we use the common way of depicting automata, where



indicates a transition $p \xrightarrow{a/b} q$.

Remark 0.2.1.1. It would actually be more precise to describe automata as finite-state, letter-to-letter **transducers** with coinciding input and output alphabets. However, the term "automaton" is the standard term used in the setting of this work.

On a few occasions, we will also talk about automata without output. In these cases, we will use the term *acceptor* to distinguish the two concepts. However, we will encounter acceptors only briefly and, thus, do not define them formally.

Properties of Automata. Let $\mathcal{T} = (Q, \Sigma, \delta)$ be an automaton. If all sets

$$\delta_{p,a} = \{p \xrightarrow{a/b} q \in \delta \mid b \in \Sigma, q \in Q\}$$

with $p \in Q$ and $a \in \Sigma$ contain at most one element, we say that $\mathcal{T}$ is *deterministic*. If they all contain at least one element, we call $\mathcal{T}$ *complete*. If all the sets

$$\bar{\delta}_{p,b} = \{p \xrightarrow{a/b} q \in \delta \mid a \in \Sigma, q \in Q\}$$

with $p \in Q$ and $b \in \Sigma$ contain at most one element, then $\mathcal{T}$ is *invertible*. Finally, $\mathcal{T}$ is *reversible* if all sets

$$\delta_{a,q} = \{p \xrightarrow{a/b} q \in \delta \mid p \in Q, b \in \Sigma\}$$

with $a \in \Sigma$ and $q \in Q$ contain at most one element (i. e. if $\mathcal{T}$ is co-deterministic with respect to the input) and it is *inverse-reversible* if all sets

$$\bar{\delta}_{b,q} = \{p \xrightarrow{a/b} q \in \delta \mid p \in Q, a \in \Sigma\}$$

with $b \in \Sigma$ and $q \in Q$ contain at most one element (i. e. if $\mathcal{T}$ is co-deterministic with respect to the output). A reversible and inverse-reversible automaton is called *bi-reversible*.

Remark 0.2.1.2. Often, bi-reversible automata are defined to also be invertible. Here, we will not make this requirement but, in most cases, we will only consider bi-reversible automata that also happen to be invertible anyway. Therefore, no confusion should arise.

Certain combinations of the above properties play an important role for the semantics of automata. Therefore, we give them special names: an $\mathscr{S}$-automaton is a deterministic (but not necessarily complete) automaton,[13] an $\overline{\mathscr{S}}$-automaton is an inverse $\mathscr{S}$-automaton and a $\mathscr{G}$-automaton is a complete $\overline{\mathscr{S}}$-automaton (see Table 0.1).

[13] We will mostly be dealing with deterministic automata. However, it is important to emphasize that we do not require $\mathscr{S}$-automata to be complete.

automaton	**properties**	**generated structure**
$\mathscr{S}$-automaton	deterministic	(partial) automaton semigroup
complete $\mathscr{S}$-automaton	deterministic, complete	complete automaton semigroup
$\overline{\mathscr{S}}$-automaton	deterministic, invertible	automaton-inverse semigroup/ inverse automaton semigroup
$\mathscr{G}$-automaton	deterministic, complete, invertible	automaton group

Table 0.1: $\mathscr{S}$-automata, $\overline{\mathscr{S}}$-automata and $\mathscr{G}$-automata compared

[14] Cross diagrams seem to have been introduced in [Akh+12, p. 1250052-16] where the authors connect them to the square diagrams of [GM05].

Cross Diagrams. We use *cross diagrams*[14] to indicate transitions in automata. To indicate that an automaton $\mathcal{T} = (Q, \Sigma, \delta)$ has a transition $p \xrightarrow{a/b} q \in \delta$, we use the cross diagram

$$\begin{array}{ccc} & a & \\ p & \downarrow & q \\ & b & \end{array} .$$

Multiple small cross diagrams can be combined into larger ones. In this way, the cross diagram

$$\begin{array}{ccccccc} & a_{0,1} & & \cdots & & a_{0,m} & \\ q_{1,0} & \downarrow & q_{1,1} & \cdots & q_{1,m-1} & \downarrow & q_{1,m} \\ & a_{1,1} & & & & a_{1,m} & \\ \vdots & \vdots & & & & \vdots & \vdots \\ & a_{n-1,1} & & & & a_{n-1,m} & \\ q_{n,0} & \downarrow & q_{n,1} & \cdots & q_{n,m-1} & \downarrow & q_{n,m} \\ & a_{n,1} & & \cdots & & a_{n,m} & \end{array}$$

states that $\mathcal{T}$ contains the transitions $q_{i,j-1} \xrightarrow{a_{i-1,j}/a_{i,j}} q_{i,j}$ for all $1 \leq i \leq n$ and $1 \leq j \leq m$. Often, we will omit unnecessary names for intermediate states and letters from cross diagrams. Additionally, we also use an abbreviated notation: the cross diagram

$$\begin{array}{ccc} & u = a_{0,1} \dots a_{0,m} & \\ \boldsymbol{p} = q_{n,0} \dots q_{1,0} & \downarrow & \boldsymbol{q} = q_{n,m} \dots q_{1,m} \\ & v = a_{n,1} \dots a_{n,m} & \end{array}$$

is an abbreviated version of the larger cross diagram given above. It is important to note the order in which the states appear in the abbreviated cross diagram: $q_{1,0}$ is the last letter of $\boldsymbol{q}$ but it is the first one in the cross diagram above. This somewhat reverse notation will makes more sense shortly; it comes from the fact that we use left actions instead of right actions.

Automaton Actions. If $\mathcal{T} = (Q, \Sigma, \delta)$ is an $\mathscr{S}$-automaton, then, for every $\boldsymbol{p} \in Q^+$ and $u \in \Sigma^+$, there is at most one $v \in \Sigma^+$ and $\boldsymbol{q} \in Q^+$ such that the cross diagram

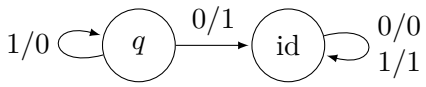
$$\begin{array}{ccc} & u & \\ \boldsymbol{p} & \downarrow & \boldsymbol{q} \\ & v & \end{array}$$

holds and we define $\boldsymbol{p} \circ u = v$ and $\boldsymbol{p} \cdot u = \boldsymbol{q}$. If no such $v \in \Sigma^+$ and $\boldsymbol{q} \in Q^+$ exist, we say that $\boldsymbol{p} \circ u$ and $\boldsymbol{p} \cdot u$ are undefined. Additionally, we let $\varepsilon \circ u = u$, $\boldsymbol{p} \circ \varepsilon = \varepsilon$, $\boldsymbol{p} \cdot \varepsilon = \boldsymbol{p}$ and $\varepsilon \cdot u = \varepsilon$. By definition, we have $\boldsymbol{qp} \circ u = \boldsymbol{q} \circ \boldsymbol{p} \circ u$ (or both undefined) and $\boldsymbol{p} \cdot uv = \boldsymbol{p} \cdot u \cdot v$ (or both undefined). Thus, we have effectively defined a partial left action of Q^* on the set Σ^* (using $\circ$), which we call the partial *action* of $\mathcal{T}$, and a partial right action of Σ^* on the set Q^* (using $\cdot$), which we call the *dual* partial *action* of $\mathcal{T}$. Since the former is

length-preserving and prefix compatible, it extends naturally into a partial action of Q^* on Σ^∞. In the same way, the latter extends into a partial action of Σ^* on $Q^* \cup Q^{-\omega}$. Usually, when we use the action notations $\boldsymbol{p} \circ u$ and $u \cdot \boldsymbol{p}$, the automaton defining these action will be clear from the context. If it is necessary to state it explicitly, we will use subscript notation, i. e. we will write $\boldsymbol{p} \circ_{\mathcal{T}} u$ and $u \cdot_{\mathcal{T}} \boldsymbol{p}$.

Remark 0.2.1.3. We will follow the convention that common word operations take precedence over the two partial automaton actions. For example $\partial \boldsymbol{q} \circ u$ means $(\partial \boldsymbol{q}) \circ u$ instead of $\partial(\boldsymbol{q} \circ u)$. Similarly, if we write $\operatorname{Suf} \boldsymbol{q} \circ u$, we mean the set $(\operatorname{Suf} \boldsymbol{q}) \circ u = \{\boldsymbol{p} \circ u \mid \boldsymbol{p} \in \operatorname{Suf} \boldsymbol{q}, \boldsymbol{p} \circ u \text{ defined}\}$ instead of the set $\operatorname{Suf}(\boldsymbol{q} \circ u)$. This allows us to use fewer parentheses.

Example 0.2.1.4 ("The Adding Machine")**.** The automaton



is known as the *adding machine*. It is deterministic, complete and invertible and, thus, a $\mathscr{G}$-automaton. However, it is neither reversible nor inverse-reversible.

The action of id on Σ^∞ is indeed the identity mapping. To understand the action of q, it is useful to first look at an example. We have the cross diagram

$$\begin{array}{ccccc}
 & 0 & 0 & 0 & \\
q & \downarrow & \downarrow & \downarrow & \\
\to & \text{id} & \to & \text{id} & \to \text{id} \\
 & 1 & 0 & 0 & \\
q & \downarrow & \downarrow & \downarrow & \\
\to & q & \to & \text{id} & \to \text{id} \\
 & 0 & 1 & 0 & \\
q & \downarrow & \downarrow & \downarrow & \\
\to & \text{id} & \to & \text{id} & \to \text{id} \\
 & 1 & 1 & 0 & \\
q & \downarrow & \downarrow & \downarrow & \\
\to & q & \to & q & \to \text{id} \\
 & 0 & 0 & 1 &
\end{array} .$$

We can interpret the input and output words in the above cross diagrams as reverse/least significant bit first binary numbers (of length three). The action on q on these binary numbers is now an increment. In fact, if we denote by $\partial \operatorname{bin} n \in \{0,1\}^\omega$ the reverse/least significant bit first binary representation of $n \in \mathbb{N}$ (of infinite length/with infinitely many trailing zeros), then we have $q^i \circ \partial \operatorname{bin} n = \partial \operatorname{bin}(n+i)$ for all $n, i \in \mathbb{N}$.

Regarding the dual action, we clearly have $\mathrm{id} \cdot u = \mathrm{id}$ for all $u \in \{0,1\}^*$ as well as $q \cdot u = \mathrm{id}$ for all $u \in \{0,1\}^* \setminus 1^*$ and $q \cdot 1^\ell = q$ for all $\ell \in \mathbb{N}$.

0.2.2 Automaton Constructions

Before we have a closer look at the semantics of automata and the algebraic structures they generate, we will first introduce some frequently used automaton constructions.

$\mathcal{T}$ is ...	$\iff$	$\partial\mathcal{T}$ is ...
deterministic		deterministic
complete		complete
invertible		reversible
inverse-reversible		inverse-reversible

Table 0.2: Properties of $\mathcal{T}$ and $\partial\mathcal{T}$

Union Automata. For two automata $\mathcal{T}_1 = (Q_1, \Sigma_1, \delta_1)$ and $\mathcal{T}_2 = (Q_2, \Sigma_2, \delta_2)$, we define their (disjoint) *union automaton* as $\mathcal{T}_1 \uplus \mathcal{T}_2 = (Q_1 \uplus Q_2, \Sigma_1 \cup \Sigma_2, \delta_1 \uplus \delta_2)$. If $\mathcal{T}_1$ and $\mathcal{T}_2$ use the same alphabet $\Sigma = \Sigma_1 = \Sigma_2$, the union automaton is deterministic/complete/invertible/reversible if $\mathcal{T}_1$ and $\mathcal{T}_2$ are both also deterministic/complete/invertible/reversible.

Composition and Power Automata. Now, let $\mathcal{T}_1 = (Q_1, \Sigma, \delta_1)$ and $\mathcal{T}_2 = (Q_2, \Sigma, \delta_2)$ be two automata over the same alphabet. Their *composition* is the automaton $\mathcal{T}_2\mathcal{T}_1 = (Q_2Q_1, \Sigma, \delta_2\delta_1)$ where $Q_2Q_1 = \{q_2q_1 \mid q_1 \in Q_1, q_2 \in Q_2\}$ is the Cartesian product of Q_2 and Q_1 and the transition are given by

$$\delta_2\delta_1 = \{p_2p_1 \xrightarrow{a/c} q_2q_1 \mid p_1 \xrightarrow{a/b} q_1 \in \delta_1, p_2 \xrightarrow{b/c} q_2 \in \delta_2\}.$$

Obviously, the composition of two deterministic/complete/invertible automata is itself deterministic/complete/invertible.

For $\mathscr{S}$-automata, the partial action of $p_2p_1 \in Q_2Q_1$ is the same regardless of whether it is seen as a state of $\mathcal{T}_2\mathcal{T}_1$ or as a state sequence over $Q_1 \uplus Q_2$, i. e. we have $p_2p_1 \circ_{\mathcal{T}_2\mathcal{T}_1} u = p_2 \circ_{\mathcal{T}_2} p_1 \circ_{\mathcal{T}_1} u$ (or both sides undefined) for all $u \in \Sigma^\infty$.

An important application of the composition of automata is the k-fold *power automaton*. For an automaton $\mathcal{T}$, we let $\mathcal{T}^1 = \mathcal{T}$ and $\mathcal{T}^k = \mathcal{T}^{k-1}\mathcal{T}$. Typically, this construction will allow us to consider state sequences $\boldsymbol{q} \in Q^k$ as single states of the power automaton while maintaining the same action.

Dual Automata. Another automaton construction is the dual automaton, in which states and letters change roles. Formally, the *dual automaton* of an automaton $\mathcal{T} = (Q, \Sigma, \delta)$ is the automaton $\partial\mathcal{T} = (\Sigma, Q, \partial\delta)$ whose transitions are given by

$$\partial\delta = \{a \xrightarrow{p/q} b \mid p \xrightarrow{a/b} q \in \delta\}.$$

Defined in this way, taking the dual automaton is an involution: $\partial\partial\mathcal{T} = \mathcal{T}$. It maintains determinism, completeness and inverse-reversibility of the automaton but it swaps the roles of invertibility and reversibility (see Table 0.2). Thus, the dual of an $\mathscr{S}$-automaton is again an $\mathscr{S}$-automaton and the dual of a bi-reversible $\mathscr{G}$-automaton is again a bi-reversible $\mathscr{G}$-automaton but the dual of a general $\mathscr{G}$-automaton is a reversible and complete $\mathscr{S}$-automaton.

To simplify our notation, we write $u \circ_\partial \boldsymbol{q}$ and $u \cdot_\partial \boldsymbol{q}$ for the partial actions of $\partial\mathcal{T}$ if the automaton is clear from the context (and if $\mathcal{T}$ is an $\mathscr{S}$-automaton). When we pass from

$\mathcal{T}$ to its dual $\partial\mathcal{T}$, we have to mirror the cross diagrams at the north-west to south-east diagonal, i. e. we have a cross diagram

$$\boldsymbol{p} \overset{u}{\underset{v}{\dashrightarrow}} \boldsymbol{q} \quad \text{in } \mathcal{T} \text{ if and only if we have the cross diagram} \quad \partial u \overset{\partial\boldsymbol{p}}{\underset{\partial\boldsymbol{q}}{\dashrightarrow}} \partial v \quad \text{in } \partial\mathcal{T}.$$

Thus. we have $\boldsymbol{p} \circ u = v \iff \partial u \cdot_\partial \partial\boldsymbol{p} = \partial v$ and $\boldsymbol{p} \cdot u = \boldsymbol{q} \iff \partial u \circ_\partial \partial\boldsymbol{p} = \partial\boldsymbol{q}$ if $\mathcal{T}$ is an $\mathscr{S}$-automaton[15] (in particular, the left hand sides are defined if and only if the right hand sides are).

[15] Recall from Remark 0.2.1.3 that ∂ has higher precedence than the partial automaton action $\circ$.

Inverse Automata. Finally, we can also take the inverse of an automaton. Let $\mathcal{T} = (Q, \Sigma, \delta)$ be an automaton. Then, its *inverse* is the automaton $\overline{\mathcal{T}} = (\overline{Q}, \Sigma, \bar{\delta})$ whose transitions are given by

$$\bar{\delta} = \{\bar{p} \xrightarrow{b/a} \bar{q} \mid p \xrightarrow{a/b} q \in \delta\}.$$

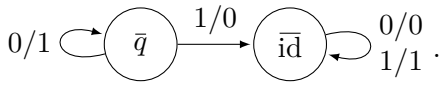
The inverse $\overline{\mathcal{T}}$ is deterministic if and only if $\mathcal{T}$ is invertible. Accordingly, the inverse of an $\overline{\mathscr{S}}$-automaton is again an $\overline{\mathscr{S}}$-automaton and the same is true for $\mathscr{G}$-automata. Additionally, taking the inverse is an involution: $\overline{\overline{\mathcal{T}}} = \mathcal{T}$. When dealing with inverse automata, we will usually also be dealing with the union $\widetilde{\mathcal{T}} = \mathcal{T} \uplus \overline{\mathcal{T}}$ of $\mathcal{T}$ and its inverse $\overline{\mathcal{T}}$.

We will not introduce additional notation for the partial actions induced by the inverse of an invertible automaton $\mathcal{T}$. Since the states $\overline{Q}$ of $\overline{\mathcal{T}}$ are disjoint to those of $\mathcal{T}$, the action is clear from the context anyway. We have a cross diagram

$$\boldsymbol{p} \overset{u}{\underset{v}{\dashrightarrow}} \boldsymbol{q} \quad \text{in } \mathcal{T} \text{ if and only if we have the cross diagram} \quad \bar{\boldsymbol{p}} \overset{v}{\underset{u}{\dashrightarrow}} \bar{\boldsymbol{q}} \quad \text{in } \overline{\mathcal{T}}.$$

In other words: when passing from $\mathcal{T}$ to $\overline{\mathcal{T}}$, we have to mirror the cross diagram horizontally and take the inverse of the state sequences. Accordingly, we have $\boldsymbol{p} \circ u = v \iff \bar{\boldsymbol{p}} \circ v = u$ or, put another way, $\bar{\boldsymbol{p}}\boldsymbol{p} \circ u = u$ for all u such that $\boldsymbol{p} \circ u$ is defined (if $\mathcal{T}$ is an $\overline{\mathscr{S}}$-automaton).

Example 0.2.2.1. The inverse of the adding machine from Example 0.2.1.4 is



The action of $\overline{\mathrm{id}}$ is the same as the action of id. The action of $\bar{q}$, on the other hand, can be considered as a decrement.

0.2.3 Automata and Algebraic Structures

We are now ready to define a few closely related concepts that play a central role in this work: we will define (complete and partial) automaton semigroups and monoids, inverse automaton semigroups and monoids, and automaton groups. As an umbrella term for these concepts, we speak of *automaton structures*.[16]

[16] A term already prominently used in the title of this work.

Automaton Semigroups and Automaton Monoids. When we have an $\mathscr{S}$-automaton $\mathcal{T} = (Q, \Sigma, \delta)$, it induces a partial action of the state sequences from Q^+ on the set of words Σ^∞. The idea is to quotient Q^+ in such a way that the action becomes faithful.

Formally, we define the relation $=_\mathcal{T} \subseteq Q^* \times Q^*$ over the state sequences by

$$\boldsymbol{p} =_\mathcal{T} \boldsymbol{q} \iff \forall w \in \Sigma^* : \boldsymbol{p} \circ w = \boldsymbol{q} \circ w \text{ (or both undefined).}$$

This is equivalent to defining it via $\boldsymbol{p} =_\mathcal{T} \boldsymbol{q} \iff \forall w \in \Sigma^\infty : \boldsymbol{p} \circ w = \boldsymbol{q} \circ w$ (or both undefined).

Clearly, $=_\mathcal{T}$ is an equivalence and it is also easy to see that it is a congruence. We will write $[\boldsymbol{q}]_\mathcal{T}$ for the congruence class of $\boldsymbol{q} \in Q^*$ with respect to $=_\mathcal{T}$. The *semigroup generated* by the $\mathscr{S}$-automaton $\mathcal{T}$ is $\mathscr{S}(\mathcal{T}) = Q^+/=_\mathcal{T}$ and a semigroup is an *automaton semigroup* if it is generated by some $\mathscr{S}$-automaton. Analogously, the *monoid generated* by $\mathcal{T}$ is $\mathscr{M}(\mathcal{T}) = Q^*/=_\mathcal{T}$ and a monoid is an *automaton monoid* if it is generated by some $\mathscr{S}$-automaton. Between the semigroup and the monoid generated by some $\mathscr{S}$-automaton, there is an obvious connection, which we will often use implicitly:

Fact 0.2.3.1. *Let $\mathcal{T} = (Q, \Sigma, \delta)$ be an $\mathscr{S}$-automaton. If there is some $\boldsymbol{q} \in Q^+$ with $\boldsymbol{q} =_\mathcal{T} \varepsilon$, then $\mathscr{M}(\mathcal{T})$ is (isomorphic to) $\mathscr{S}(\mathcal{T})$. If there is no such state sequence, then $\mathscr{M}(\mathcal{T})$ is (isomorphic to) $\mathscr{S}(\mathcal{T})^1$.*

The Functional View. Sometimes, it will be more useful to consider automaton semigroups and monoids as subsemigroups of $\mathscr{P}(\Sigma^*) = \{f \mid f : \Sigma^* \rightharpoonup \Sigma^*\}$ or of $\mathscr{P}(\Sigma^\infty) = \{f \mid f : \Sigma^\infty \rightharpoonup \Sigma^\infty\}$.

If $\mathcal{T} = (Q, \Sigma, \delta)$ is an $\mathscr{S}$-automaton, every state sequence $\boldsymbol{q} \in Q^*$ induces a partial function $\boldsymbol{q}\circ : \Sigma^\infty \rightharpoonup \Sigma^\infty$ mapping $u \in \Sigma^\infty$ to $\boldsymbol{q} \circ u$. Since these functions are given by a partial action, the composition of $\boldsymbol{q}\circ$ with $\boldsymbol{p}\circ$ is $\boldsymbol{qp}\circ$ and, since they are prefix-compatible, two maps $\boldsymbol{p}\circ : \Sigma^\infty \rightharpoonup \Sigma^\infty$ and $\boldsymbol{q}\circ : \Sigma^\infty \rightharpoonup \Sigma^\infty$ coincide if and only if their restrictions into partial maps $\Sigma^* \rightharpoonup \Sigma^*$ do. We can take the closure $Q^+\circ = \{\boldsymbol{q}\circ \mid \boldsymbol{q} \in Q^+\}$ of these maps under composition, which forms a subsemigroup of $\mathscr{P}(\Sigma^\infty)$ (or $\mathscr{P}(\Sigma^*)$, respectively). It is not difficult to see that this subsemigroup is isomorphic to $\mathscr{S}(\mathcal{T})$. Similarly, the monoid generated by $\mathcal{T}$ is isomorphic to $Q^*\circ = \{\boldsymbol{q}\circ \mid \boldsymbol{q} \in Q^*\}$.

Complete Automaton Semigroups and Complete Automaton Monoids. If an automaton semigroup is generated by a complete $\mathscr{S}$-automaton, we call it a *complete* automaton semigroup and the definition of *complete* automaton monoids is analogous. This does not only reflect the completeness of the generating automaton: if we consider a complete automaton semigroup or monoid under the functional view, then all maps $\boldsymbol{q}\circ$ with $\boldsymbol{q} \in Q^*$ are in fact total maps $\Sigma^\infty \to \Sigma^\infty$.

Remark 0.2.3.2. In the literature, automaton semigroups are usually defined only using complete automata, i. e., there, the term "automaton semigroup" implies that the generating automaton is complete. We will discuss the motivation to also consider partial automata and some questions about the difference between these two concepts in Section 1.1.

The term "automaton monoid", on the other hand, is not widely used in the literature anyway. Sometimes the term "automaton semigroup" even refers to the generated monoid. In this work, however, we will make a clear distinction between the two concepts.

Inverse Automaton Semigroups and Inverse Automaton Monoids. The functional view of automaton semigroups and monoids comes in particularly handy if we consider invertible automata. If $\mathcal{T} = (Q, \Sigma, \delta)$ is an $\overline{\mathscr{S}}$-automaton, then every function $\boldsymbol{q}\circ$ with $\boldsymbol{q} \in Q^*$ is one-to-one. This implies that the semigroup or monoid generated by $\mathcal{T}$ (under the functional view) is a (not necessarily inverse) subsemigroup of the symmetric inverse semigroup $\mathscr{I}(\Sigma^\infty)$ (or $\mathscr{I}(\Sigma^*)$). If we now also consider the inverse automaton, then we obtain that the semigroup generated by $\widetilde{\mathcal{T}} = \mathcal{T} \uplus \overline{\mathcal{T}}$ is an inverse semigroup. We call it the *inverse semigroup generated* by $\mathcal{T}$ and denote it by $\overline{\mathscr{S}}(\mathcal{T}) = \mathscr{S}(\widetilde{\mathcal{T}})$. A semigroup is an *automaton-inverse semigroup* if it is the inverse semigroup generated by some $\overline{\mathscr{S}}$-automaton. In the same way, the *inverse monoid generated* by $\mathcal{T}$ is $\overline{\mathscr{M}}(\mathcal{T}) = \mathscr{M}(\widetilde{\mathcal{T}})$ and any monoid generated in this way is an *automaton-inverse monoid*. Note that, a priori, there is a difference between automaton-inverse semigroups or monoids and automaton semigroups or monoids that happen to be inverse (see Example 1.3.2.1). However, we will show in Theorem 1.3.2.6, that both concepts describe the same class of semigroups or monoids. Therefore, we will usually use the simpler terms "inverse automaton semigroup" and "inverse automaton monoid" unless the distinction is of importance.

Remark 0.2.3.3. We will discuss inverse automaton semigroups and monoids in more detail in Section 1.3. There, the reader may also find some examples of $\overline{\mathscr{S}}$-automata and their generated inverse semigroups and monoids.

Automaton Groups. If we take a complete $\mathscr{S}$-automaton $\mathcal{T} = (Q, \Sigma, \delta)$, then all functions $\boldsymbol{q}\circ$ with $\boldsymbol{q} \in Q^*$ are total. If the $\mathscr{S}$-automaton is invertible, then they are one-to-one. If we combine these two properties, i. e. if $\mathcal{T}$ is a $\mathscr{G}$-automaton, we obtain that the functions are actually bijections $\Sigma^\infty \to \Sigma^\infty$ (or $\Sigma^* \to \Sigma^*$). Thus, in this case, the inverse monoid $\overline{\mathscr{M}}(\mathcal{T})$ generated by $\mathcal{T}$ (under the functional view) is a group or, more precisely, it is a subgroup of the symmetric group over Σ^∞ (or Σ^*). Additionally, we have that $\overline{\mathscr{S}}(\mathcal{T})$ is (isomorphic to) $\overline{\mathscr{M}}(\mathcal{T})$ because we have $\bar{q}q =_{\widetilde{\mathcal{T}}} \varepsilon$ for any $q \in Q$. To emphasize that we are dealing with a group, we write $\mathscr{G}(\mathcal{T})$ for $\overline{\mathscr{M}}(\mathcal{T})$ or $\overline{\mathscr{S}}(\mathcal{T})$ if $\mathcal{T}$ is a $\mathscr{G}$-automaton and say that $\mathscr{G}(\mathcal{T})$ is the *group generated* by $\mathcal{T}$. Accordingly, an *automaton group* is a group generated by some $\mathscr{G}$-automaton.

Remark 0.2.3.4. As with inverse automaton semigroups, there is a difference in the definition between automaton groups and automaton semigroups that happen to be groups. However, it turns out that also these two classes coincide. For the case of complete automaton semigroups, this has already been shown by Cain [Cai09, Proposition 3.1] and, for the case of partial automata, we will show this in Corollary 1.3.2.7.

Example 0.2.3.5. Recall from Example 0.2.1.4 that the action of q in the adding machine $\mathcal{T}$ can be considered as an increment on reverse/least significant bit first binary numbers: $q^i \circ \partial \operatorname{bin} n = \partial \operatorname{bin}(n+i)$ for all $i, n \in \mathbb{N}$. Thus, we have $q^i \neq_{\mathcal{T}} q^j$ for $i \neq j$. Additionally, we have $q \operatorname{id} =_{\mathcal{T}} \operatorname{id} q =_{\mathcal{T}} q$. Thus, the semigroup generated by the adding

machine is (isomorphic to) the monogenic free monoid q^*. Since id acts as the identity (i. e. since we have $\mathrm{id} =_{\mathcal{T}} \varepsilon$), the monoid generated by $\mathcal{T}$ coincides with (or, more precisely, is isomorphic to) the semigroup generated by $\mathcal{T}$. This turns the monogenic free monoid q^* into a complete automaton semigroup (and a complete automaton monoid).

The (semigroup) generators of the group $\mathscr{G}(\mathcal{T})$ under the functional view are the increment $q\circ$, the identity $\mathrm{id}\circ$and the decrement $\bar{q}\circ$, where we have $q\bar{q}\circ = \bar{q}q\circ = \mathrm{id}\circ = \varepsilon\circ$, which means that $\mathscr{G}(\mathcal{T})$ is the free group in one generator.

0.3 Automaton Actions

0.3.1 The Dual Action

We have used the action given by an $\mathscr{S}$-automaton to define algebraic structures. In this subsection, we will have a closer look at the dual partial action given by an $\mathscr{S}$-automaton.

Dual Action on an Automaton Semigroup. The dual $\partial\mathcal{T}$ of an $\mathscr{S}$-automaton $\mathcal{T} = (Q, \Sigma, \delta)$ is also an $\mathscr{S}$-automaton and, thus, generates a semigroup. This semigroup, however, is not an algebraic property of $\mathscr{S}(\mathcal{T})$ but rather a property of the presentation via $\mathcal{T}$. However, the dual partial action of Σ^* on Q^* induced by $\mathcal{T}$ is compatible with the structure of $\mathscr{S}(\mathcal{T})$ and $\mathscr{M}(\mathcal{T})$.

Fact 0.3.1.1. *Let $\mathcal{T} = (Q, \Sigma, \delta)$ be an $\mathscr{S}$-automaton and let $\boldsymbol{p}, \boldsymbol{q} \in Q^*$ with $\boldsymbol{p} =_{\mathcal{T}} \boldsymbol{q}$. Then, we have $\boldsymbol{p} \cdot w =_{\mathcal{T}} \boldsymbol{q} \cdot w$ for all $w \in \Sigma^*$.*

Proof. For $w = \varepsilon$, there is nothing to show. If $w = w'a$ for some $a \in \Sigma$, then we have $\boldsymbol{p}' = \boldsymbol{p} \cdot w' =_{\mathcal{T}} \boldsymbol{q} \cdot w' = \boldsymbol{q}'$ by induction and we have to show $\boldsymbol{p}' \cdot a =_{\mathcal{T}} \boldsymbol{q}' \cdot a$. Let $u \in \Sigma^*$ be arbitrary. We have $(\boldsymbol{p}' \circ a)(\boldsymbol{p}' \cdot a \circ u) = \boldsymbol{p}' \circ au = \boldsymbol{q}' \circ au = (\boldsymbol{q}' \circ a)(\boldsymbol{q}' \cdot a \circ u)$ and, thus, $\boldsymbol{p}' \cdot a \circ u = \boldsymbol{q}' \cdot a \circ u$. □

This allows us to extend our notations for the dual partial action of Σ^* on Q^* to elements of $\mathscr{S}(\mathcal{T})$ and $\mathscr{M}(\mathcal{T})$: for $\boldsymbol{q} \in Q^*$ and $u \in \Sigma^*$, we let $[\boldsymbol{q}]_{\mathcal{T}} \cdot u = [\boldsymbol{q} \cdot u]_{\mathcal{T}}$ (if $\boldsymbol{q} \cdot u$ is defined; otherwise, we let $[\boldsymbol{q}]_{\mathcal{T}} \cdot u$ be undefined as well). This way, we can write $s \cdot u$ for an abstract element $s \in \mathscr{S}(\mathcal{T})$.

The Functional View of the Dual. We can extend the idea of the functional view to the dual partial action. For an $\mathscr{S}$-automaton $\mathcal{T} = (Q, \Sigma, \delta)$, every finite word $w \in \Sigma^*$ partially acts on $Q^* \cup Q^{-\omega}$. Thus, it induces a partial function $\cdot w : Q^* \cup Q^{-\omega} \rightharpoonup Q^* \cup Q^{-\omega}$, $\boldsymbol{q} \mapsto \boldsymbol{q} \cdot w$. By the definition of the dual automaton, $\cdot w$ can be obtained from $w \circ_\partial$ by taking the composition with the reversing operator ∂ and we can observe that $\cdot w$ is suffix-compatible and length-preserving.

If $\mathcal{T}$ is complete, the functions $\cdot w$ with $w \in \Sigma^*$ are total and, just like the functions $\boldsymbol{q} \circ$ with $\boldsymbol{q} \in Q^*$ are permutations of Σ^∞ if the automaton is invertible and complete (i. e. a $\mathscr{G}$-automaton), the functions $\cdot w$ with $w \in \Sigma^*$ are permutations of $Q^* \cup Q^{-\omega}$ if the automaton is reversible and complete. More precisely, we have:

Fact 0.3.1.2. *If $\mathcal{T} = (Q, \Sigma, \delta)$ is a complete and reversible $\mathscr{S}$-automaton, all functions $\cdot\, w$ with $w \in \Sigma^*$ mapping $\boldsymbol{q}$ to $\boldsymbol{q} \cdot w$ are length-preserving, suffix-compatible permutations of $Q^* \cup Q^{-\omega}$.*

Proof. The fact follows immediately if we consider the dual automaton: we have $\boldsymbol{q} \cdot w = \partial(\partial w \circ_\partial \partial \boldsymbol{q})$ and $\partial w \circ_\partial$ is a length-preserving, prefix-compatible permutation of Q^∞ since $\partial\mathcal{T}$ is a $\mathscr{G}$-automaton. □

0.3.2 Stabilizers, Orbits and Orbital Graphs

Stabilizers. For an $\mathscr{S}$-automaton $\mathcal{T} = (Q, \Sigma, \delta)$ and a word $u \in \Sigma^\infty$, the sets

$$\mathrm{Stab}_\mathcal{T}(u) = \{\boldsymbol{q} \in Q^+ \mid \boldsymbol{q} \circ u = u \text{ (defined, in particular)}\}$$

and

$$\mathrm{Stab}^{\mathbb{1}}_\mathcal{T}(u) = \{\boldsymbol{q} \in Q^* \mid \boldsymbol{q} \circ u = u \text{ (defined, in particular)}\} = \mathrm{Stab}_\mathcal{T}(u) \cup \{\varepsilon\}$$

contain the state sequences which *stabilize* u. Obviously, $\mathrm{Stab}_\mathcal{T}(u)$ and $\mathrm{Stab}^{\mathbb{1}}_\mathcal{T}(u)$ are closed under product, i. e., if one of them contains $\boldsymbol{p}$ and $\boldsymbol{q}$, it will also contain $\boldsymbol{pq}$. Thus, the image of $\mathrm{Stab}_\mathcal{T}(u)$ in $\mathscr{S}(\mathcal{T})$ forms a subsemigroup of $\mathscr{S}(\mathcal{T})$ and $\mathrm{Stab}^{\mathbb{1}}_\mathcal{T}(u)$ forms a submonoid in $\mathscr{M}(\mathcal{T})$. If $\mathcal{T}$ is invertible, then $\mathrm{Stab}_{\widetilde{\mathcal{T}}}(u)$ is closed under taking the inverse, i. e. we have $\bar{\boldsymbol{q}} \in \mathrm{Stab}_{\widetilde{\mathcal{T}}}(u)$ for all $\boldsymbol{q} \in \widetilde{Q}^+$ with $\boldsymbol{q} \in \mathrm{Stab}_{\widetilde{\mathcal{T}}}(u)$. Accordingly, $\mathrm{Stab}^{\mathbb{1}}_{\widetilde{\mathcal{T}}}(u)$ forms a subgroup in $\mathscr{G}(\mathcal{T})$ if $\mathcal{T}$ is a $\mathscr{G}$-automaton. These substructures are called the (semigroup, monoid and group) *stabilizer* of u (respectively).

K-Orbits, Orbits and Orbital Graphs. Let $\mathcal{T} = (Q, \Sigma, \delta)$ be an $\mathscr{S}$-automaton. For $K \subseteq Q^*$, the *K-orbit* of a word $w \in \Sigma^\infty$ (under the action of $\mathcal{T}$) is the set

$$K \circ w = \{\boldsymbol{q} \circ w \mid \boldsymbol{q} \in K, \boldsymbol{q} \circ w \text{ defined}\}.$$

The Q^*-orbit of a word w is simply called the *orbit* of w.

The orbit of a word has a natural graph structure: the *orbital graph* of a word $w \in \Sigma^\infty$ is a directed graph whose node set is $Q^* \circ w$ and whose edges are given by

$$\{q \circ u \xleftarrow{q} u \mid u \in Q^* \circ w, q \circ u \text{ defined}\}.$$

To avoid notational overhead we will use $Q^* \circ w$ to denote both the orbit of w and the orbital graph of w. It will be clear from the context whether we refer to the orbit as a set or to the orbital graph. Clearly, the orbital graph $Q^* \circ w$ is connected, in the sense that every node can be reached from its *initial node* w.

Remark 0.3.2.1. Because we have defined automaton semigroups using a *left* action, our orbital graphs are somewhat "backwards". For example, (if defined) the node $q_n \dots q_1 \circ w$ with $q_1, \dots, q_n \in Q$ can be reached from w by a path whose first edge is labeled by q_1 and whose last edge is labeled by q_n. Thus, it is sometimes useful to write paths (and edges) in orbital graphs from right to left, i. e. we denote the path mentioned above by

$$q_n \dots q_1 \circ w \xleftarrow{q_n} \cdots \xleftarrow{q_2} q_1 \circ w \xleftarrow{q_1} w\ .$$

However, at other times, we will also write paths (and edges) in the normal way from left to right.

Orbits of Invertible Automata. If $\mathcal{T} = (Q, \Sigma, \delta)$ is invertible, we can also take its inverse into consideration. In this case, the orbital graph $\widetilde{Q}^* \circ w$ of a word $w \in \Sigma^\infty$ (under the action of $\widetilde{\mathcal{T}} = \mathcal{T} \uplus \overline{\mathcal{T}}$) is an undirected graph in the sense that, for every edge $v \xleftarrow{q} u$, we also have the back edge $v \xrightarrow{\bar{q}} u$.

In the case of a $\mathscr{G}$-automaton, it turns out that the orbits $Q^* \circ w$ and $\widetilde{Q}^* \circ w$ are either equal (if they are finite) or they are both infinite. This is a well-known connection that we state in the following fact. In fact, we extend it here to finitely generated subgroups.

Fact 0.3.2.2 (generalization of [3, Lemma 2.5])**.** *Let $\mathcal{T} = (Q, \Sigma, \delta)$ be a $\mathscr{G}$-automaton and let $\boldsymbol{P} \subseteq \widetilde{Q}^*$ be finite.*

Then, for any $w \in \Sigma^\infty$, we have

$$\boldsymbol{P}^* \circ w = \widetilde{\boldsymbol{P}}^* \circ w \quad \text{or} \quad |\boldsymbol{P}^* \circ w| = \left|\widetilde{\boldsymbol{P}}^* \circ w\right| = \infty.$$

Proof. By taking the union with suitable powers of $\mathcal{T}$, we can, without loss of generality, assume $\boldsymbol{P} = P \subseteq \widetilde{Q}$. By definition, we have $P^* \circ w \subseteq \widetilde{P}^* \circ w$. Thus, if $P^* \circ w$ is infinite, $\widetilde{P}^* \circ w$ must also be infinite.

Thus, let $P^* \circ w$ be finite. Every $p \in P$ induces a mapping $P^* \circ w \to P^* \circ w, \boldsymbol{q} \circ w \mapsto p\boldsymbol{q} \circ w$. This mapping is total because $\mathcal{T}$ is complete and it is injective because $\mathcal{T}$ is invertible. Since $P^* \circ w$ is of finite size, the mapping must also be surjective for reasons of cardinality and is, thus, a permutation of the finite set $P^* \circ w$. Therefore, there must be some natural number[17] $\pi \geq 1$ such that we have $p^\pi \circ u = u$ for all $u \in P^* \circ w$. We obtain $\bar{p} \circ u = \bar{p} p^\pi \circ u = p^{\pi-1} \circ u$ for all $u \in P^* \circ w$. This means that every node in the orbital graph $\widetilde{Q}^* \circ w$ that can be reached from w by a path with a label in $\widetilde{P}^*$ can also be reached from w in $Q^* \circ w$ by a path with a label in P^* because we can replace edges labeled by $\bar{p}$ by paths labeled by $p^{\pi-1}$. □

[17] In fact, π is the order of the map as an element of the symmetric group over $P^* \circ w$.

The completeness of the automaton is essential to the last proof. In fact, also considering the inverse can indeed increase the orbit size for (non-complete) $\overline{\mathscr{S}}$-automata.

Counter Example 0.3.2.3 (compare to [3, Lemma 2.6])**.** Let $\mathcal{T} = (Q, \Sigma, \delta)$ be a $\mathscr{G}$-automaton such that there is some[18] $\alpha \in \Sigma^\omega$ with $|Q^* \circ \alpha| = |\widetilde{Q}^* \circ \alpha| = \infty$. Without loss of generality,[19] we may assume that there is some state $q_1 \in Q$ whose action is the identity on Σ^∞. Let $Q' = \{q' \mid q \in Q\}$ be a disjoint copy of Q and let $\hat{\Sigma} = \{\hat{a} \mid a \in \Sigma\}$ be a disjoint copy of Σ. We define $\mathcal{T}' = (Q', \Sigma \uplus \hat{\Sigma}, \delta')$ as the automaton with the transitions

$$\delta' = \{p' \xrightarrow{a/\hat{b}} q' \mid p \xrightarrow{a/b} q \in \delta\},$$

i. e. every state acts in the same way as before but, additionally, adds a hat decoration to every output letter. Certainly, $\mathcal{T}'$ is still invertible but not complete anymore. We have that $q' \circ w$ is undefined for any $w \in (\Sigma \uplus \hat{\Sigma})^\infty$ that contains at least one letter from $\hat{\Sigma}$ and for all $q' \in Q'$. Thus, the partial action of any state sequence from $(Q')^*$ containing more than a single state is undefined on all words (except the empty one). Therefore, we have $|(Q')^* \circ w| \leq |Q'| + 1 = |Q| + 1$ for all $w \in (\Sigma \uplus \hat{\Sigma})^\infty$. A symmetric argument shows that $|(\overline{Q}')^* \circ w|$ must also be bounded by $|Q| + 1$. Thus, all purely positive orbits $(Q')^* \circ w$ and all purely negative orbits $(\overline{Q}')^* \circ w$ are finite.

[18] We will see later in Subsection 1.4.1 that such a word exists if and only if $\mathscr{G}(\mathcal{T})$ is infinite.

[19] If no such state exists, we can simply add it to the automaton without altering the generated group.

On the other hand, the action of $\overline{q_1}'$ is to remove the hat decoration from every letter. Therefore, the partial action of $\overline{q_1}'q'$ is the same as the action of $q \in Q$ on words from Σ^∞. Accordingly, we have $Q^* \circ \alpha \subseteq (\widetilde{Q}')^* \circ \alpha$ and, thus, that $(\widetilde{Q}')^* \circ \alpha$ is of infinite size.

L-Dual Orbits and Dual Orbits. An $\mathscr{S}$-automaton $\mathcal{T} = (Q, \Sigma, \delta)$ also induces the dual partial action of Σ^* on $Q^* \cup Q^{-\omega}$, for which we can define an orbit as well. For a language $L \subseteq \Sigma^*$, the *L-dual orbit* of a sequence $\boldsymbol{q} \in Q^* \cup Q^{-\omega}$ (with respect to $\mathcal{T}$) is

$$\boldsymbol{q} \cdot L = \{\boldsymbol{q} \cdot u \mid u \in L, \boldsymbol{q} \cdot u \text{ defined}\}.$$

Symmetrically to (normal) orbits, the Σ^*-dual orbit of $\boldsymbol{q}$ is simply called the *dual orbit* of $\boldsymbol{q}$.

The dual orbit of a state sequence is closely related to its orbit under the action of the dual; in fact, they are in bijection by taking the reverse. This can easily be seen as we only have to mirror the cross diagrams if we pass to the dual automaton.

Fact 0.3.2.4. *Let $\mathcal{T} = (Q, \Sigma, \delta)$ be an $\mathscr{S}$-automaton and let $\boldsymbol{q} \in Q^* \cup Q^{-\omega}$. The map*

$$\begin{aligned} \boldsymbol{q} \cdot \Sigma^* &\to \Sigma^* \circ_\partial \partial\boldsymbol{q} \\ \boldsymbol{q} \cdot u &\mapsto \partial(\boldsymbol{q} \cdot u) = \partial u \circ_\partial \partial\boldsymbol{q} \end{aligned}$$

is a bijection.

By Fact 0.3.1.1, the $\mathscr{S}$-automaton $\mathcal{T}$ also induces a (dual) partial action of Σ^* on $\mathscr{S}(\mathcal{T})$. This allows us to also define the dual orbit of a semigroup element $s \in \mathscr{S}(\mathcal{T})$. It is

$$s \cdot \Sigma^* = \{s \cdot u \mid u \in \Sigma^*, s \cdot u \text{ defined}\}.$$

1 Structure Theory

1.1 Partial and Complete Automata

With our definition, an automaton semigroup is generated by a partial and, thus, possibly non-complete automaton. In the literature, the notion of automaton semigroups is often defined using complete automata only. However, there are reasons for allowing partial instead of only complete automata. First of all, it is a natural notion: while group actions need to be total to maintain the group notion of invertibility, there is no such requirement for semigroups. In fact, partial actions are essential for the study of inverse semigroups and, using partial automata, we are able to define the notion of inverse automaton semigroups (which we will discuss more thoroughly in Section 1.3). Inverse automaton semigroups are interesting because they bridge the gap between automaton semigroups and automaton groups. This idea of bridging the gap has turned out to be also interesting algorithmically as we will see later on in Section 2.1 and Section 2.3.

Additionally, there does not seem to be a good reason not to take possibly non-complete automata into consideration. Usually, it turns out that a result holding for complete automaton semigroups can also be shown for (partial) automaton semigroups. In fact, we will see in this section that we can show certain closure properties for the class of (partial) automaton semigroups that are unknown for the class of complete automaton semigroups.

Every semigroup acts faithfully on itself (after possibly adjoining an identity element) and, thus, every semigroup element can be represented as a total function from some set to the same set. This includes semigroups that actually consist of partial functions. Therefore, for general semigroups, it does not seem to be a restriction to only consider total functions and one can ask whether the same is true for automaton semigroups: does the class of (partial) automaton semigroups coincide with the class of complete automaton semigroups? As every complete automaton semigroup is in particular also a partial one, we can also formulate this question as:

Open Problem 1.1.0.1. Is every (partial) automaton semigroup a complete automaton semigroup?

It is this question that motivates our study in this section. First, we will see that we can go from a (partial) automaton semigroup to a complete one by re-using an existing left zero or by adjoining a new zero. Then, we will see that the class of (partial) automaton semigroups is closed under removing a previously adjoined zero. This is interesting because it answers the analogue of a question asked by Cain [Cai09, Open problem 5.3] on complete automaton semigroups for the class of partial automaton semigroups. Additionally, it shows that an adjoined zero can always be assumed to be of a certain form in the generating automaton; more precisely, it can always be assumed

to be the partial function that is undefined everywhere (except for ε). The relation between this closure property and Cain's question finally leads to a partial answer to Open Problem 1.1.0.1: the classes coincide if and only if Cain's question has a positive answer as we will see at the end of this section.

Attribution. The results in this section are joint work with Daniele D'Angeli and Emanuele Rodaro [9] and the presentation mostly follows that work (with minor changes to the proofs). The only exception is Proposition 1.1.1.1, which is generalized to use left zeros instead of zeros. The idea to pass from a (partial) automaton semigroup to a complete one by adjoining a zero is already stated in [1, Proposition 1] and a small oversight in its proof led to some more general results given here (see [9] for a short discussion of this). Some of the shown constructions are inspired by similar ones given by Cain [Cai09] and [Cai09, Open problem 5.3] is of particular interest to our discussion.

Other results on partial automaton structures seem only to exist in the context of inverse semigroups, which we discuss further in Section 1.3.

1.1.1 Re-Using and Adjoining Zeros

Our first result is that we can use an existing zero in an automaton semigroup to complete the generating automaton. In fact, it suffices to have a left zero.

Proposition 1.1.1.1 (extension of [9, Proposition 6] to left zeros)**.** *Every (partial) automaton semigroup with a left zero is a complete automaton semigroup.*

Proof. Let $S = \mathscr{S}(\mathcal{T})$ for some (partial) $\mathscr{S}$-automaton $\mathcal{T} = (Q, \Sigma, \delta)$ contain a left zero. Without loss of generality, we may assume that there is some $z \in Q$ which is this left zero in S (otherwise we can substitute $\mathcal{T}$ by $\mathcal{T} \uplus \mathcal{T}^i$ for a suitable i). We will define a complete $\mathscr{S}$-automaton $\hat{\mathcal{T}} = (\hat{Q}, \hat{\Sigma}, \hat{\delta})$ such that $\hat{S} = \mathscr{S}(\hat{\mathcal{T}})$ is isomorphic to S. Let $\hat{Q}$ be a disjoint copy of Q. Notice that we have $\hat{z} \in \hat{Q}$. We extend the notation $\hat{q}$ to finite words over Q by defining $\hat{\boldsymbol{q}} = \hat{q}_n \dots \hat{q}_1$ for all $\boldsymbol{q} = q_n \dots q_1 \in Q^*$ (with $q_1, \dots, q_n \in Q$). For the alphabet, we choose $\hat{\Sigma} = \Sigma \uplus \{\bot\}$ for a new symbol $\bot$. Finally, the transitions of $\hat{\mathcal{T}}$ are given by

$$\begin{aligned}\hat{\delta} = \{\hat{p} \xrightarrow{a/b} \hat{q} \mid p \xrightarrow{a/b} q \in \delta\} \cup \\ \{\hat{p} \xrightarrow{a/\bot} \hat{z} \mid p \cdot a \text{ undefined}\} \cup \{\hat{p} \xrightarrow{\bot/\bot} \hat{z} \mid p \in Q\},\end{aligned}$$

i. e. we basically have the transitions of $\mathcal{T}$ and some additional ones to make $\hat{\mathcal{T}}$ complete. We will show that mapping $\boldsymbol{q}$ to $\hat{\boldsymbol{q}}$ for all $\boldsymbol{q} \in Q^+$ induces a well-defined isomorphism $\iota : S \to \hat{S}$.

To show that ι is well-defined, we have to show that $\boldsymbol{p} = \boldsymbol{q}$ in S implies $\hat{\boldsymbol{p}} = \hat{\boldsymbol{q}}$ in $\hat{S}$ for all $\boldsymbol{p}, \boldsymbol{q} \in Q^+$. We show the latter by showing $\hat{\boldsymbol{p}} \circ w = \hat{\boldsymbol{q}} \circ w$ for all $\boldsymbol{p}, \boldsymbol{q} \in Q^+$ and all $w \in \hat{\Sigma}^*$ using an induction on the length of w. For $w = \varepsilon$, there is nothing to show. Therefore, let $w = au$ for some $a \in \hat{\Sigma}$.

The first case is that $a = \bot \notin \Sigma$. By construction of $\hat{\mathcal{T}}$, we have the cross diagrams

$$\begin{array}{ccc} & \bot & \\ \hat{\boldsymbol{p}} & \downarrow & \hat{z}^{|\boldsymbol{p}|} \\ & \bot & \end{array} \quad \text{and} \quad \begin{array}{ccc} & \bot & \\ \hat{\boldsymbol{q}} & \downarrow & \hat{z}^{|\boldsymbol{q}|} \\ & \bot & \end{array} .$$

Since z is a left zero in S, we have $z^{|\boldsymbol{p}|} = z = z^{|\boldsymbol{q}|}$ in S. Thus, we obtain $\hat{\boldsymbol{p}} \circ \bot u = \bot(\hat{z}^{|\boldsymbol{p}|} \circ u) = \bot(\hat{z}^{|\boldsymbol{q}|} \circ u) = \hat{\boldsymbol{q}} \circ \bot u$ from induction.

Next, let $a \in \Sigma$ and $\boldsymbol{p} \circ a = \boldsymbol{q} \circ a$ (both) defined. Thus, if we write $\boldsymbol{p} = p_m \dots p_1$ and $\boldsymbol{q} = q_n \dots q_1$ for $p_1, \dots, p_m, q_1, \dots, q_n \in Q$, we have the cross diagrams

$$\begin{array}{ccc} & a & \\ p_1 & \to & p_1' \\ \vdots & \vdots & \vdots \\ p_m & \to & p_m' \\ & b & \end{array} \quad \text{and} \quad \begin{array}{ccc} & a & \\ q_1 & \to & q_1' \\ \vdots & \vdots & \vdots \\ q_n & \to & q_n' \\ & b & \end{array}$$

for some $p_1', \dots, p_m', q_1', \dots, q_n' \in Q$ and $b \in \Sigma$. By the construction of $\hat{\mathcal{T}}$, this yields the cross diagrams

$$\begin{array}{ccc} & a & \\ \hat{p}_1 & \to & \hat{p}_1' \\ \vdots & \vdots & \vdots \\ \hat{p}_n & \to & \hat{p}_n' \\ & b & \end{array} \quad \text{and} \quad \begin{array}{ccc} & a & \\ \hat{q}_1 & \to & \hat{q}_1' \\ \vdots & \vdots & \vdots \\ \hat{q}_n & \to & \hat{q}_m' \\ & b & \end{array} .$$

From $p_n' \dots p_1' = q_m' \dots q_1'$ in S, we obtain $\hat{\boldsymbol{p}} \circ au = b(\hat{p}_n' \dots \hat{p}_1' \circ u) = b(\hat{q}_m' \dots \hat{q}_1' \circ u) = \hat{\boldsymbol{q}} \circ au$ by induction.

Finally, let $a \in \Sigma$ and let $\boldsymbol{p} \circ a$ and $\boldsymbol{q} \circ a$ both be undefined. We can factorize $\boldsymbol{p} = \boldsymbol{p}_3 p_2 \boldsymbol{p}_1$ for $\boldsymbol{p}_1, \boldsymbol{p}_3 \in Q^*$ and $p_2 \in Q$ such that $\boldsymbol{p}_1 \circ a$ is defined but $p_2 \boldsymbol{p}_1 \circ a$ is not. Factorizing $\boldsymbol{q}$ in an analogous way, we obtain the cross diagrams

$$\begin{array}{ccc} & a & \\ \hat{\boldsymbol{p}}_1 & \to & \hat{\boldsymbol{p}}_1' \\ & b & \\ \hat{p}_2 & \to & \hat{z} \\ & \bot & \\ \hat{\boldsymbol{p}}_3 & \to & \hat{z}^{|\boldsymbol{p}_3|} \\ & \bot & \end{array} \quad \text{and} \quad \begin{array}{ccc} & a & \\ \hat{\boldsymbol{q}}_1 & \to & \hat{\boldsymbol{q}}_1' \\ & c & \\ \hat{q}_2 & \to & \hat{z} \\ & \bot & \\ \hat{\boldsymbol{q}}_3 & \to & \hat{z}^{|\boldsymbol{q}_3|} \\ & \bot & \end{array}$$

for $\boldsymbol{p}_1' = \boldsymbol{p}_1 \cdot a$, $\boldsymbol{q}_1' = \boldsymbol{q}_1 \cdot a$ and some $b, c \in \Sigma$ from the construction of $\hat{\mathcal{T}}$ (similarly to the cross diagrams above). Because z is a left zero in S, we have $z^{|\boldsymbol{p}_3|} z \boldsymbol{p}_1' = z = z^{|\boldsymbol{q}_3|} z \boldsymbol{q}_1'$ in S and obtain $\hat{\boldsymbol{p}} \circ au = \bot(\hat{z}^{|\boldsymbol{p}_3|} \hat{z} \hat{\boldsymbol{p}}_1' \circ u) = \bot(\hat{z}^{|\boldsymbol{q}_3|} \hat{z} \hat{\boldsymbol{q}}_1' \circ u) = \hat{\boldsymbol{q}} \circ au$ from induction.

By definition, ι is both a homomorphism and surjective. Therefore, it only remains to show that ι is injective. Assume that we have $\boldsymbol{p} \neq \boldsymbol{q}$ in S for some $\boldsymbol{p}, \boldsymbol{q} \in Q^+$. This can only be the case if there is some witness $u \in \Sigma^*$ such that either $\boldsymbol{p} \circ u$ and $\boldsymbol{q} \circ u$ are both defined but their values differ or one of them (say: $\boldsymbol{p} \circ u$) is defined while the other ($\boldsymbol{q} \circ u$) is not. In the first case, we have $\hat{\boldsymbol{p}} \circ u = \boldsymbol{p} \circ u \neq \boldsymbol{q} \circ u = \hat{\boldsymbol{q}} \circ u$ and, in the second case, we have that $\hat{\boldsymbol{p}} \circ u = \boldsymbol{p} \circ u$ does not contain a $\bot$ while $\hat{\boldsymbol{q}} \circ u$ does. In either case, we have $\hat{\boldsymbol{p}} \neq \hat{\boldsymbol{q}}$ in $\hat{S}$. □

We continue by showing that we can adjoin zeros to semigroups without leaving the class of (partial) automaton semigroups.[20] This is the analogue of [Cai09, Proposition 5.1] for partial automaton semigroups.

[20] The idea is similar to the one in the proof of [Cai09, Proposition 5.2].

Proposition 1.1.1.2 (see [9, Proposition 7])**.** *For all semigroups S, we have:*

$$S \text{ is a (partial) automaton semigroup} \implies S^{\mathbb{0}} \text{ is a (partial) automaton semigroup}$$

Proof. Let $S = \mathscr{S}(\mathcal{T})$ for some $\mathscr{S}$-automaton $\mathcal{T} = (Q, \Sigma, \delta)$. Let $\hat{Q}$ be a disjoint copy of Q and let $\hat{\boldsymbol{q}} = \hat{q}_n \dots \hat{q}_1$ for all $\boldsymbol{q} = q_n \dots q_1$ with $q_1, \dots, q_n \in Q$. We define the $\mathscr{S}$-automaton $\hat{\mathcal{T}} = (\hat{Q} \uplus \{z\}, \Sigma \uplus \{\top\}, \hat{\delta})$ where z is a new state and $\top$ is a new letter with the transitions

$$\hat{\delta} = \{\hat{p} \xrightarrow{a/b} \hat{q} \mid p \xrightarrow{a/b} q \in \delta\} \cup \{\hat{p} \xrightarrow{\top/\top} \hat{p} \mid p \in Q\},$$

i. e. we basically have the transitions from $\mathcal{T}$ and, additionally, loops at every state mapping $\top$ to $\top$. In particular, there are no transitions from z or to z and, therefore, $z \circ u$ is undefined on all $u \in (\Sigma \uplus \{\top\})^*$ except the empty word. This makes z a zero in $\hat{T} = \mathscr{S}(\hat{\mathcal{T}})$. We also have $\hat{\boldsymbol{q}} \circ \top = \top$ for all $\boldsymbol{q} \in Q^+$ while $z \circ \top$ is undefined. Thus, we have $\hat{\boldsymbol{q}} \neq z$ in $\hat{T}$ for all $\boldsymbol{q} \in Q^+$. We obtain that $\hat{S}$, the semigroup generated by $\hat{Q}$ in $\hat{T}$, contains all elements in $\hat{T}$ except z. In other words, we have $\hat{T} = \hat{S}^{\mathbb{0}}$.

We claim that mapping $\boldsymbol{q}$ to $\hat{\boldsymbol{q}}$ for all $\boldsymbol{q} \in Q^*$ induces an isomorphism $\iota : S \to \hat{S}$, which will conclude our argument. To show that ι is well-defined, we use an induction on the length of $w \in (\Sigma \uplus \{\top\})^*$ and show that $\boldsymbol{p} = \boldsymbol{q}$ in S implies $\boldsymbol{p} \circ w = \boldsymbol{q} \circ w$ (or both undefined) for all $\boldsymbol{p}, \boldsymbol{q} \in Q^+$.

For $w = \varepsilon$, there is nothing to show. If $w = \top u$ for some $u \in (\Sigma \uplus \{\top\})^*$, then it is easy to see that we have $\hat{\boldsymbol{p}} \circ \top u = \top(\hat{\boldsymbol{p}} \circ u) = \top(\hat{\boldsymbol{q}} \circ u) = \hat{\boldsymbol{q}} \circ \top u$ (or all undefined) where the equality in the middle follows by induction. Finally, let $w = au$ for some $a \in \Sigma$ and some $u \in (\Sigma \uplus \{\top\})^*$. If $\boldsymbol{p} \circ a$ is undefined, then $\hat{\boldsymbol{q}} \circ a$ is, too, and the same holds for $\boldsymbol{q} \circ a$ and $\hat{\boldsymbol{q}} \circ a$. Therefore, in this case, we have that $\hat{\boldsymbol{p}} \circ au$ and $\hat{\boldsymbol{q}} \circ au$ are both undefined. If $\boldsymbol{p} \circ a = \boldsymbol{q} \circ a$ are both defined, then it is easy to see from the construction of $\hat{\mathcal{T}}$ that we have $\hat{\boldsymbol{p}} \circ au = (\hat{\boldsymbol{p}} \circ a)(\hat{\boldsymbol{p}}' \circ u) = (\boldsymbol{p} \circ a)(\hat{\boldsymbol{p}}' \circ u)$ (or all undefined) where $\boldsymbol{p}' = \boldsymbol{p} \cdot a$. In the same way, we also obtain $\hat{\boldsymbol{q}} \circ au = (\boldsymbol{q} \circ a)(\hat{\boldsymbol{q}}' \circ u)$ (or both undefined) for $\boldsymbol{q}' = \boldsymbol{q} \cdot a$. Since we have $\boldsymbol{p}' = \boldsymbol{q}'$ in S, we are done by induction.

It remains to show that ι is injective, surjective and a homomorphism. However, the latter two are trivial and, for the former, we observe that a witness $u \in \Sigma^*$ for $\boldsymbol{p} \neq \boldsymbol{q}$ in S is also a witness for $\hat{\boldsymbol{p}} \neq \hat{\boldsymbol{q}}$ in $\hat{S}$. □

Combining Proposition 1.1.1.1 and Proposition 1.1.1.2, we can obtain a complete automaton semigroup from any (partial) automaton semigroup by adjoining a zero.

Corollary 1.1.1.3 (see [1, Proposition 1] and [9, Corollary 8])**.** *For all semigroups S, we have:*

$$S \text{ is a (partial) automaton semigroup} \implies S^{\mathbb{0}} \text{ is a complete automaton semigroup}$$

1.1.2 Removing Zeros

We have seen that we can use an existing (left) zero to go from a partial automaton semigroup to a complete one. In this section, we will see that we can safely remove a previously adjoined zero from a (partial) automaton semigroup.

First, we define a construction on $\mathscr{S}$-automata that does not change the generated semigroup. Later on, this construction will allow us to remove some transitions from the automaton without coincidentally creating more equalities in the semigroup.

Definition 1.1.2.1 (end marker extension, see [9, Definition 11])**.** Let $\mathcal{T} = (Q, \Sigma, \delta)$ be an automaton. Define the disjoint copy $\Sigma_\$ = \{a_\$ \mid a \in \Sigma\}$ of Σ, whose elements are called *end marker letters*. The *end marker extension* of $\mathcal{T}$ is the automaton $\hat{\mathcal{T}} = (\hat{Q}, \hat{\Sigma}, \hat{\delta})$ where $\hat{Q}$ is a disjoint copy of Q, $\hat{\Sigma}$ is the alphabet $\hat{\Sigma} = \Sigma \uplus \Sigma_\$$ and the transitions $\hat{\delta}$ are given by

$$\hat{\delta} = \{\hat{p} \xrightarrow{a/b} \hat{q} \mid p \xrightarrow{a/b} q \in \delta\} \cup \{\hat{p} \xrightarrow{a_\$/b_\$} \hat{p} \mid p \xrightarrow{a/b} q \in \delta \text{ for some } q \in Q\}.$$

So, the end marker extension of $\mathcal{T}$ has the same transitions as $\mathcal{T}$ and additional self-loops for the end marker letters. It is easy to see that the end marker extension of an $\mathscr{S}$-automaton is again an $\mathscr{S}$-automaton.[21]

[21] Other properties such as completeness and invertibility are maintained as well, of course. However, we will only need that the construction preserves determinism.

First, we show that adding these self-loops does not change the generated semigroup. This allows us to assume that any automaton semigroup is generated by an end marker extension automaton.

Proposition 1.1.2.2 (see [9, Lemma 12])**.** *If $\hat{\mathcal{T}}$ is the end marker extension of some $\mathscr{S}$-automaton $\mathcal{T}$, then $\mathscr{S}(\hat{\mathcal{T}})$ and $\mathscr{S}(\mathcal{T})$ are isomorphic.*

Proof. Let $\mathcal{T} = (Q, \Sigma, \delta)$ and $\hat{\mathcal{T}} = (\hat{Q}, \hat{\Sigma}, \hat{\delta})$. For any state sequence $\boldsymbol{p} = p_n \dots p_1 \in Q^*$ with $p_1, \dots, p_n \in Q$, let $\hat{\boldsymbol{p}} = \hat{p}_n \dots \hat{p}_1$. We will show that mapping $\boldsymbol{p}$ to $\hat{\boldsymbol{p}}$ defines a well-defined isomorphism $\iota : \mathscr{S}(\mathcal{T}) \to \mathscr{S}(\hat{\mathcal{T}})$.

To show that it is well-defined, we will show, for all $w \in \hat{\Sigma}^*$ and all $\boldsymbol{p}, \boldsymbol{q} \in Q^+$, that $\boldsymbol{p} = \boldsymbol{q}$ in $\mathscr{S}(\mathcal{T})$ implies $\hat{\boldsymbol{p}} \circ w = \hat{\boldsymbol{q}} \circ w$ (or both undefined) and we do this using an induction on the length of w. For $w = \varepsilon$, there is nothing to show. Suppose that we have $w = au$ for some $a \in \Sigma$ and $u \in \hat{\Sigma}^*$ and $\boldsymbol{p} = \boldsymbol{q}$ in $\mathscr{S}(\mathcal{T})$. If $\boldsymbol{p} \circ a$ and $\boldsymbol{q} \circ a$ are both undefined, then we also have that $\hat{\boldsymbol{p}} \circ a$ and $\hat{\boldsymbol{q}} \circ a$ are both undefined (by the construction of the end marker extension). Thus, let $\boldsymbol{p} \circ a = \boldsymbol{q} \circ a = b$ be defined. By construction of $\hat{\mathcal{T}}$, the cross diagram

$$\boldsymbol{p} \overset{a}{\underset{b}{\downarrow\!\rightarrow}} \boldsymbol{p}' \quad \text{in } \mathcal{T} \text{ implies the cross diagram} \quad \hat{\boldsymbol{p}} \overset{a}{\underset{b}{\downarrow\!\rightarrow}} \hat{\boldsymbol{p}}' \quad \text{in } \hat{\mathcal{T}}.$$

Using an analogous argument and analogous definitions for $\boldsymbol{q}$, we obtain $\hat{\boldsymbol{p}} \circ au = b(\hat{\boldsymbol{p}}' \circ u) = b(\hat{\boldsymbol{q}}' \circ u) = \hat{\boldsymbol{q}} \circ au$ (or all of them undefined) from induction since we must have $\boldsymbol{p}' = \boldsymbol{q}'$ in $\mathscr{S}(\mathcal{T})$.

The remaining case is that we have $w = a_\$ u$ for some $a \in \Sigma$ and $u \in \hat{\Sigma}^*$ and, again, $\boldsymbol{p} = \boldsymbol{q}$ in $\mathscr{S}(\mathcal{T})$. We show this case using a contradiction and assume that we have $\hat{\boldsymbol{p}} \circ a_\$ u \neq \hat{\boldsymbol{q}} \circ a_\$ u$ (including the case that one side is undefined while the other one is not).

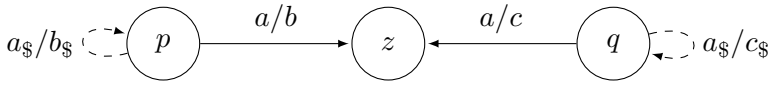


Figure 1.1: Removing z without introducing the end marker self-loops causes q and p to become the same element in the generated semigroup ([9, Fig. 1]).

Without loss of generality, we assume that $\hat{\boldsymbol{p}} \circ a_\$ u$ is defined. This can only be the case if $\boldsymbol{p} \circ a = \boldsymbol{q} \circ a = b$ is defined. By construction, we, therefore, have the cross diagrams

$$\hat{\boldsymbol{p}} \overset{a_\$}{\underset{b_\$}{\dashrightarrow}} \hat{\boldsymbol{p}} \quad \text{and} \quad \hat{\boldsymbol{q}} \overset{a_\$}{\underset{b_\$}{\dashrightarrow}} \hat{\boldsymbol{q}} \quad \text{in } \hat{\mathcal{T}}$$

and, thus, $\hat{\boldsymbol{p}} \circ a_\$ u = b_\$(\hat{\boldsymbol{p}} \circ u) = b_\$(\hat{\boldsymbol{q}} \circ u) = \hat{\boldsymbol{q}} \circ a_\$ u$ (or all undefined) by induction.

That ι is surjective is trivial and that it is injective is clear since $\hat{\boldsymbol{p}} = \hat{\boldsymbol{q}}$ in $\mathscr{S}(\hat{\mathcal{T}})$ implies $\boldsymbol{p} = \boldsymbol{q}$ in $\mathscr{S}(\mathcal{T})$ as we just have to restrict the alphabet to go from $\hat{\boldsymbol{p}}$ to $\boldsymbol{p}$ and from $\hat{\boldsymbol{q}}$ to $\boldsymbol{q}$. □

Now, we come to the main point for this section: the class of (partial) automaton semigroups is closed under removing an adjoined zero. The basic idea of the proof for this is that we take the generating automaton and remove all states that represent the zero in the semigroup. In general, this might change the generated semigroup (see Figure 1.1) but it turns out that this does not happen if we pass to the end marker extension before removing the zeros.

Proposition 1.1.2.3 (see [9, Proposition 13])**.** *For all semigroups S, we have:*

$$S^{\mathbb{0}} \text{ is a (partial) automaton semigroup} \implies S \text{ is a (partial) automaton semigroup}$$

Proof. Let S be a semigroup such that $S^{\mathbb{0}}$ is generated by some $\mathscr{S}$-automaton $\mathcal{T} = (Q, \Sigma, \delta)$. By Proposition 1.1.2.2, we may safely assume that $\mathcal{T}$ is an end marker extension (of some other automaton). Define $Z = \{\boldsymbol{z} \in Q^+ \mid \boldsymbol{z} \text{ is the zero in } \mathscr{S}(\mathcal{T})\}$ and let $Q' = \{q' \mid q \in Q \setminus Z\}$ be a copy of $Q \setminus Z$. We define the automaton $\mathcal{T}' = (Q', \Sigma, \delta')$ via the transitions

$$\delta' = \{p' \xrightarrow{a/b} q' \mid p \xrightarrow{a/b} q \in \delta, p, q \notin Z\}.$$

In other words, the automaton $\mathcal{T}'$ basically contains the same transitions as $\mathcal{T}$ except those that go into or come from a state belonging to Z. We will show that $\mathcal{T}'$ generates $S = S^{\mathbb{0}} \setminus \{\mathbb{0}\}$. For this, we first define the notation $\boldsymbol{p}' = p'_n \dots p'_1$ for all $\boldsymbol{p} = p_n \dots p_1$ with $p_1, \dots, p_n \in Q \setminus Z$ and claim that mapping $\boldsymbol{p}'$ to $\boldsymbol{p}$ for all $\boldsymbol{p} \in (Q \setminus Z)^+$ induces a well-defined, injective homomorphism $\iota : \mathscr{S}(\mathcal{T}') \to \mathscr{S}(\mathcal{T}') = S^{\mathbb{0}}$ whose image is S.

Before we show this claim, we observe that a cross diagram

$$\boldsymbol{p}' \overset{u}{\underset{v}{\dashrightarrow}} \boldsymbol{r}' \quad \text{in } \mathcal{T}' \text{ yields the cross diagram} \quad \boldsymbol{p} \overset{u}{\underset{v}{\dashrightarrow}} \boldsymbol{r} \quad \text{in } \mathcal{T}$$

by the construction of $\mathcal{T}'$. In particular, we have that $\boldsymbol{p} \circ u$ is defined for some $\boldsymbol{p} \in (Q \setminus Z)^+$ if $\boldsymbol{p}' \circ u$ is. In fact, we have $\boldsymbol{p} \circ u = \boldsymbol{p}' \circ u$ in this case. By contraposition, this means that $\boldsymbol{p}' \circ u$ must be undefined if $\boldsymbol{p} \circ u$ is.

Now, we show that ι is well-defined by showing that, for all $w \in \Sigma^*$ and all $\boldsymbol{p}', \boldsymbol{q}' \in (Q')^+$, $\boldsymbol{p}' = \boldsymbol{q}'$ in $\mathscr{S}(\mathcal{T}')$ implies $\boldsymbol{p} \circ w = \boldsymbol{q} \circ w$ using an induction on the length of w. For $w = \varepsilon$, there is nothing to show. Thus, let $w = au$ for some $a \in \Sigma$ and $u \in \Sigma^*$. If $\boldsymbol{p}' \circ a = \boldsymbol{q}' \circ a = b$ is defined, we have the cross diagrams

$$\boldsymbol{p}' \overset{a}{\underset{b}{\dashrightarrow}} \boldsymbol{r}' \quad \text{and} \quad \boldsymbol{q}' \overset{a}{\underset{b}{\dashrightarrow}} \boldsymbol{s}' \quad \text{in } \mathcal{T}' \text{ for some } \boldsymbol{r}', \boldsymbol{s}' \in (Q')^+.$$

This yields the analogous cross diagrams with $\boldsymbol{p}$ and $\boldsymbol{r}$ or $\boldsymbol{p}$ and $\boldsymbol{s}$, respectively, in $\mathcal{T}$. Therefore, we have $\boldsymbol{p} \circ au = b(\boldsymbol{r} \circ u) = b(\boldsymbol{s} \circ u) = \boldsymbol{q} \circ au$ (or all undefined) by induction since we must have $\boldsymbol{r}' = \boldsymbol{s}'$ in $\mathscr{S}(\mathcal{T}')$.

We show the case where $\boldsymbol{p}' \circ a$ and $\boldsymbol{q}' \circ a$ are both undefined using a contradiction and assume that we have $\boldsymbol{p}' = \boldsymbol{q}'$ in $\mathscr{S}(\mathcal{T}')$ but $\boldsymbol{p} \circ au \neq \boldsymbol{q} \circ au$ (including the case that one is defined while the other one is not). Without loss of generality, we may assume that $\boldsymbol{p} \circ au$ (and, thus, also $\boldsymbol{p} \circ a$) is defined. Remember that $\mathcal{T}$ is an end marker extension. If a is not already an end marker letter, then we can consider the corresponding $a_\$$. On the other hand, if a is an end marker letter, then a is of the form $a_\$$ anyway. Since $\boldsymbol{p} \circ a$ is defined, we have that $\boldsymbol{p} \circ a_\$ = b_\$$ must be defined as well and we obtain the cross diagrams

$$\boldsymbol{p} \overset{a_\$}{\underset{b_\$}{\dashrightarrow}} \boldsymbol{p} \quad \text{in } \mathcal{T} \text{ and} \quad \boldsymbol{p}' \overset{a_\$}{\underset{b_\$}{\dashrightarrow}} \boldsymbol{p}' \quad \text{in } \mathcal{T}'$$

(since $\boldsymbol{p}$ cannot contain a state from Z).

If $\boldsymbol{q} \circ a$ is undefined, we have that also $\boldsymbol{q} \circ a_\$$ must be undefined (by the construction of end marker extensions). Thus, we have that $\boldsymbol{q}' \circ a_\$$ is undefined while $\boldsymbol{p}' \circ a_\$ = b_\$$ is defined; a contradiction. Similarly, if $\boldsymbol{q} \circ a = c$ is defined but we have $b \neq c$, we have $\boldsymbol{q}' \circ a_\$ = c_\$ \neq b_\$ = \boldsymbol{p}' \circ a_\$$ and a contradiction as well.

Finally, let $b = \boldsymbol{p} \circ a = \boldsymbol{q} \circ a$ be (both) defined. Since $\boldsymbol{p}' \circ a$ and $\boldsymbol{q}' \circ a$ are both undefined, there must be a state z from Z in $\boldsymbol{p} \cdot a$ and in $\boldsymbol{q} \cdot a$. Since this state is the zero in $\mathscr{S}(\mathcal{T})$, we obtain that $\boldsymbol{p} \cdot a = \boldsymbol{q} \cdot a = z \in Z$ in $\mathscr{S}(\mathcal{T})$. Thus, we have $\boldsymbol{p} \circ au = b(z \circ u) = \boldsymbol{q} \circ au$ (or all undefined), which contradicts our assumption $\boldsymbol{p} \circ au \neq \boldsymbol{q} \circ au$.

This concludes the proof that ι is well-defined and it remains to show that it is injective and that its image is S. We first show the latter. Since S is a subsemigroup of $S^{\mathbb{0}} = \mathscr{S}(\mathcal{T})$, no image $\boldsymbol{q} \in (Q \setminus Z)^+$ of an element $\boldsymbol{q}' \in (Q')^+$ can be in Z. On the other hand, for every element $s \in S$ there is some $\boldsymbol{q} \in Q^+$ that is s in $\mathscr{S}(\mathcal{T})$. No such $\boldsymbol{q}$ can contain a state in Z as this would mean that $\boldsymbol{q}$ is the zero in $\mathscr{S}(\mathcal{T})$, which is not in S. Therefore, $\boldsymbol{q}'$ is a preimage of $\boldsymbol{q}$ under ι.

Finally, we show that ι is injective by showing that, for all $w \in \Sigma^*$ and all $\boldsymbol{p}, \boldsymbol{q} \in (Q \setminus Z)^+$, we have that $\boldsymbol{p} = \boldsymbol{q}$ in $\mathscr{S}(\mathcal{T})$ implies $\boldsymbol{p}' \circ w = \boldsymbol{q}' \circ w$ (or both undefined). We do this

by an induction on the length of w. For $w = \varepsilon$, there is nothing to show. Therefore, let $w = au$ for some $a \in \Sigma$ and $u \in \Sigma^*$. If $\boldsymbol{p}' \circ a$ and $\boldsymbol{q}' \circ a$ are both undefined, then we are done as also $\boldsymbol{p}' \circ au$ and $\boldsymbol{q}' \circ au$ must be undefined. Thus, we assume without loss of generality that $\boldsymbol{p}' \circ a = b$ is defined. This yields the cross diagrams

$$\boldsymbol{p}' \overset{a}{\underset{b}{\dashrightarrow}} \boldsymbol{r}' \quad \text{in } \mathcal{T}' \text{ as well as } \quad \boldsymbol{p} \overset{a}{\underset{b}{\dashrightarrow}} \boldsymbol{r} \quad \text{and} \quad \boldsymbol{q} \overset{a}{\underset{b}{\dashrightarrow}} \boldsymbol{s} \quad \text{in } \mathcal{T}$$

for some $\boldsymbol{r}' \in (Q')^+$ and $\boldsymbol{s} \in Q^+$ with $\boldsymbol{s} = \boldsymbol{r}$ in $\mathscr{S}(\mathcal{T})$. Note that $\boldsymbol{r}$ cannot contain a state from Z (since it is given by $\boldsymbol{r}'$) and that, thus, $\boldsymbol{r}$ is not the zero in $\mathscr{S}(\mathcal{T})$. Therefore, $\boldsymbol{s}$ cannot contain a state from Z either. By the construction of $\mathcal{T}'$, this yields the cross diagram

$$\boldsymbol{q}' \overset{a}{\underset{b}{\dashrightarrow}} \boldsymbol{s}' \quad \text{in } \mathcal{T}'.$$

Thus, we have $\boldsymbol{p}' \circ au = b(\boldsymbol{r}' \circ u) = b(\boldsymbol{s}' \circ u) = \boldsymbol{q}' \circ au$ (or all undefined) by induction. □

We can combine Proposition 1.1.1.2 and Proposition 1.1.2.3 to obtain the following closure property for the class of (partial) automaton semigroups.

Theorem 1.1.2.4. *For all semigroups S, we have:*

$$S^{\mathbb{0}} \text{ is a (partial) automaton semigroup} \iff S \text{ is a (partial) automaton semigroup}$$

If we combine Proposition 1.1.2.3 and (the proof of) Proposition 1.1.1.2 in a different way, we can say something about the structure of adjoined zeros in the generating automaton:

Proposition 1.1.2.5. *For every (partial or complete) automaton semigroup of the form $S^{\mathbb{0}}$, there is some $\mathscr{S}$-automaton $\mathcal{T} = (Q, \Sigma, \delta)$ with $\mathscr{S}(\mathcal{T})$ isomorphic to $S^{\mathbb{0}}$ such that the action of the zero is undefined on all $u \in \Sigma^*$ except for ε.*

1.1.3 A Problem Due to Cain

We have seen that the class of (partial) automaton semigroups is closed under adding and removing adjoined zeros in Theorem 1.1.2.4. At first, the underlying question seems a bit artificial. However, it turns that it is not only closely related to the very natural question whether the classes of (partial) automaton semigroups and complete automaton semigroups coincide (Open Problem 1.1.0.1) but also to a similar open problem given by Cain [Cai09, Open problem 5.3], that arose without considering non-complete automata:

Open Problem 1.1.3.1 ([Cai09, Open problem 5.3], see also [9, Problem 9])**.** Does

$$S^{\mathbb{0}} \text{ is a complete automaton semigroup} \implies S \text{ is a complete automaton semigroup}$$

hold for all semigroups S?

In fact, Open Problem 1.1.3.1 is asking whether the analogue of Theorem 1.1.2.4 for the class of complete automaton semigroups holds (since it is closed under adjoining a zero by Corollary 1.1.1.3 or [Cai09, Proposition 5.1]). Thus, if the two classes coincide, i. e., if Open Problem 1.1.0.1 has a positive answer, then also Open Problem 1.1.3.1 has a positive answer.

On the other hand, if Open Problem 1.1.3.1 has a positive answer, we can start with a (partial) automaton semigroup S, adjoin a zero to obtain a complete automaton semigroup (by Corollary 1.1.1.3) and then use the positive answer to finally obtain that S must be a complete automaton semigroup. This shows the following equivalence.

Proposition 1.1.3.2. *Open Problem 1.1.3.1 has a positive answer if and only if Open Problem 1.1.0.1 has.*

1.2 Non-Automaton Semigroups

We have discussed the problem of whether the class of (partial) automaton semigroups coincides with that of complete automaton semigroups in Section 1.1. If we want to show that the two classes are distinct, we need to disprove that some (partial) automaton semigroup is a complete automaton semigroup. However, there do not seem to exist general techniques for this.

One approach to disprove that some semigroup is an automaton semigroup, is to show that the semigroup in question does not satisfy certain properties common to all automaton semigroups (e. g. being finitely generated, residually finite [Cai09, Proposition 3.2] or having a word problem in PSpace, see Section 2.1). In our setting, however, this approach does not seem to be very useful because the typical properties of complete automaton semigroups are also shared by all (partial) automaton semigroups.

The number of example semigroups that are not (complete or partial) automaton semigroups but share the typical properties of automaton semigroups seems to be very limited. One example was given by Cain [Cai09, Proposition 4.3]: the monogenic free semigroup q^+ is not a complete automaton semigroup.[22] Later this example was generalized by Brough and Cain [BC17, Theorem 15] who showed that no (non-empty and) non-trivial subsemigroup of $(q^+)^{\mathbb{0}}$ is a complete automaton semigroup.

In this section, we will further extend this result in two ways: first, we will also consider partial automata and, second, we show that no semidirect product of an arbitrary semigroup and q^+ is an automaton semigroup. In fact, we will make a slightly more general statement (about subsemigroups of small extensions of q^+) that truly generalizes Brough and Cain's result.

While, unfortunately, our result does not help with the original question about the relation of the class of (partial) automaton semigroups and the class of complete automaton semigroups, it still substantially increases the class of known non-automaton semigroups and gives more insight into possible methods for disproving that a semigroup is not a (complete or partial) automaton semigroup.

[22] On the other hand, the monogenic free monoid (Example 0.2.3.5) and the free (or free commutative) semigroups of higher rank are complete automaton semigroups [Cai09, Proposition 4.1] (or [Cai09, Proposition 4.2] respectively).

Attribution. The results presented here generalize the corresponding results of [9], which is joint work with Daniele D'Angeli and Emanuele Rodaro. In particular, considering ideal small extensions (as a generalization of zeros in a semigroup) is a novel approach. The overall proof, however, is still based on the core idea originally presented by Cain [Cai09, Theorem 15] and later elaborated on by Brough and Cain [BC17, Theorem 15].

1.2.1 Semidirect Products of the Monogenic Free Semigroup

Ideal Small Extensions. Let S be a semigroup. A subset $I \subseteq S$ is a *right ideal* if $\{is \mid i \in I, s \in S\} = IS \subseteq I$ holds; analogously, it is a *left ideal* if $\{si \mid s \in S, i \in I\} = SI \subseteq I$ holds. A *(two-sided) ideal* is a subset I that is both a left and a right ideal. In this case, we have $SIS \subseteq I$. Notice that, in particular, $\emptyset$ is an ideal of any semigroup and that any (left or right) ideal is a subsemigroup.

A semigroup T is a *small extension* of S if S is a subsemigroup of T and the *Rees index* $|T - S|$ of S in T is finite. An *ideal small extension* of S is a small extension T of S such that $T - S$ is an ideal of T.

Example 1.2.1.1. $S^{\mathbb{0}}$ is an ideal small extension of S for any semigroup S. More generally, if S is an arbitrary semigroup and T is a finite semigroup, then $S \uplus T$ with $st = ts = t$ for all $s \in S$ and $t \in T$ (in addition to the product in S and in T) is an ideal small extension of S.

Nearly Injective Maps. A function $f : A \to B$ from a set A to a set B is *nearly injective* if there is a constant C such that $f^{-1}(b) = \{a \mid f(a) = b\}$ contains at most C elements for all $b \in B$.

We are mostly interested in nearly injective homomorphisms between semigroups and start by observing that an element has torsion if and only if its image under a nearly injective homomorphism has torsion.

Fact 1.2.1.2 ([9, Lemma 16])**.** *Let S and T be semigroups and let $\gamma : S \to T$ be a nearly injective homomorphism. Then, $s \in S$ has torsion in S if and only if $\gamma(s)$ has torsion in T.*

Proof. Let $t = \gamma(s)$ for some $s \in S$. Then, t has torsion in T if and only if the subsemigroup $t^+ = \{t^i \mid i \geq 1\}$ of T is finite. If this is the case, then $\gamma^{-1}(t^+) \supseteq s^+$ must also be finite since γ is nearly injective. This is equivalent to s having torsion. □

We will use the notion of nearly injective homomorphisms to generalize the following idea. Suppose we have some state q of an $\mathscr{S}$-automaton and all out-going transitions from q are self-loops back to q (i. e. q *recurses* only to itself), then the action of q has torsion. This basic idea also underlies [BC17, Lemma 14] where the element may also recurse to a zero. Here, we will generalize this further to the case that all reachable elements are either mapped into some finite ideal or to the same element under some nearly injective homomorphism. A typical application of the following lemma is, thus, that the additional ideal consists of a single zero or is entirely empty.

Lemma 1.2.1.3 (see also [9, Lemma 17]). *Let $\mathcal{T} = (Q, \Sigma, \delta)$ be an $\mathscr{S}$-automaton such that there is a nearly injective homomorphism $\gamma : S \to T$ from $S = \mathscr{S}(\mathcal{T})$ to some (arbitrary) semigroup T. Furthermore, let I be a finite ideal of T and $s \in S$.*

If $\gamma(s \cdot \Sigma^*) \setminus I$ contains at most one element, then s has torsion.[23]

Proof. Let $t = \gamma(s)$. By hypothesis, we already have $\gamma(s \cdot \Sigma^*) \subseteq I \cup \{t\}$ and will first show that we have $\gamma(s^i \cdot \Sigma^*) \subseteq I \cup \{t^i\}$ for all $i \geq 1$. Let $i \geq 1$ and $w \in \Sigma^*$ such that $s^i \cdot w$ is defined. Then, we have the cross diagram[24]

$$\begin{array}{ccc} & w_0 = w & \\ s & \longrightarrow & s \cdot w_0 \\ & \downarrow & \\ & w_1 & \\ s & \longrightarrow & s \cdot w_1 \\ & \vdots & \\ & w_{i-1} & \\ s & \longrightarrow & s \cdot w_{i-1}. \\ & \downarrow & \\ & w_i & \end{array}$$

for some $w_0, \ldots, w_{i-1} \in \Sigma^*$. If there is some $0 \leq k < i$ with $\gamma(s \cdot w_k) \in I$, we have $\gamma(s^i \cdot w) = \gamma(s \cdot w_0) \ldots \gamma(s \cdot w_k) \ldots \gamma(s \cdot w_{i-1}) \in TIT \subseteq I$. By hypothesis, the only other option is $\gamma(s \cdot w_k) = t$ for all $0 \leq k < i$. In this case, we have $\gamma(s^i \cdot w) = t^i$.

Since γ is nearly injective, there is some constant C such that $\gamma^{-1}(t')$ contains at most C elements for all $t' \in T$. With $\gamma(s^i \cdot \Sigma^*) \subseteq I \cup \{t^i\}$, this implies $|s^i \cdot \Sigma^*| \leq (|I|+1)C = K$ for all $i \geq 1$. In other words, each s^i can be defined using an $\mathscr{S}$-automaton over Σ of size at most K. Since there are only finitely many (non-isomorphic) such automata, we obtain $s^i = s^j$ for some $i, j \geq 1$ with $i \neq j$. □

[23] For the notation $s \cdot \Sigma^*$, recall that the partial action of Σ^* on Q^* also induces a well-defined partial action of Σ^* on $\mathscr{S}(\mathcal{T})$ by Fact 0.3.1.1, which allowed us to extend the notation for dual orbits to semigroup elements.

[24] The cross diagram uses semigroup elements instead of state sequences (with the obvious semantics). This is also well-defined by Fact 0.3.1.1.

Semidirect Products of Semigroups. Let T be a semigroup acting on some other semigroup S (from the left), i. e. there is a homomorphism $\alpha : T \to \mathrm{End}(S)$, $t \mapsto \alpha_t$ where $\mathrm{End}(S)$ is the endomorphism monoid of S with composition as operation. The *semidirect product* $S \rtimes_\alpha T$ is the semigroup with elements $(s, t) \in S \times T$ and the multiplication $(s, t)(s', t') = (s\alpha_t(s'), tt')$ (where we multiply in S in the first component and in T in the second). We simply write $S \rtimes T$ when the action of T on S is given implicitly.

We will show that no (infinite subsemigroup of a) semidirect product of the monogenic free semigroup q^+ is an automaton semigroup. This extends the result by Brough and Cain [BC17, Theorem 15] (based on [Cai09, Proposition 4.3]) that no subsemigroup of $(q^+)^0$ is an automaton semigroup to a wider class of semigroups. The central proof idea, however, is still that of [Cai09, Proposition 4.3].

Theorem 1.2.1.4 (generalization of [9, Theorem 19]). *Let S be an arbitrary but non-empty semigroup and let T be an infinite subsemigroup of an ideal small extension of q^+. Then, $S \rtimes T$ is not an automaton semigroup.*

Proof. Let $\hat{T}$ be the ideal small extension of q^+ of which T is an infinite subsemigroup. Then, $\hat{I} = \hat{T} - q^+$ is a finite ideal of $\hat{T}$ and $I = \hat{I} \cap T$ is a finite ideal of T. Since T is

infinite, there must be some element of the form q^i for some $i \geq 1$ in T. To simplify our notation, we write such elements as i and use additive instead of multiplicative notation, i. e. we write $i + j$ instead of $q^i q^j$ for these elements.

Since $T \setminus I$ is non-empty, we can define $\ell = \min T \setminus I$ (where the minimum is taken with respect to the ordering of the natural numbers) and observe that all (s, ℓ) with $s \in S$ must be in any generating set for $S \rtimes T$. Therefore, if S is infinite, we are done since $S \rtimes T$ is not finitely generated and, thus, no automaton semigroup in this case.

We show the case where S is finite using a contradiction. Therefore, assume that $S \rtimes T = \mathscr{S}(\mathcal{T})$ for some $\mathscr{S}$-automaton $\mathcal{T} = (Q, \Sigma, \delta)$. If we denote the action of ℓ on S by $f \in \mathrm{End}(S)$, then there has to be some $k \geq 1$ with $f^k = f^{2k}$ (where f^k is the k-fold composition of f). Furthermore, if we denote by R the image of Q in $\mathscr{S}(\mathcal{T})$ and by $R_2 = \{j \mid (r, j) \in R\}$, then $R_2 \setminus I$ is finite (since Q and R are) but also non-empty (since R contains all (s, ℓ) with $s \in S$). Thus, the maximum $L = \max R_2 \setminus I$ exists and there is some $s \in S$ with $(s, L) \in R$. Also note that all (s', ℓ) with $s' \in S$ must be in R (as mentioned above) and that $r \in R^j$ implies $r \cdot \Sigma^* \subseteq R^j$ for all j.

Without loss of generality, we may assume that L is a multiple of $k\ell$: since (s, L) is in R, its power $(s, L)^{k\ell} = (s', k\ell L)$ is the image of a state of $\mathcal{T}^{k\ell}$ and its second component $k\ell L$ is not in I. Furthermore, $k\ell L$ is the largest value among all second components of such elements. Thus, if L is not a multiple of $k\ell$, we can replace $\mathcal{T}$ by $\mathcal{T} \uplus \mathcal{T}^{k\ell}$.

The action of L on S is $f^{\frac{L}{\ell}} = f^{k\lambda} = f^k$ (for $\lambda = \frac{L}{k\ell}$) and we can calculate the powers $(s, L)^j$ for $j \geq 2$:

$$\begin{aligned}(s, L)^j &= (s, L)^{j-2}(s, L)(s, L) = (s, L)^{j-3}(s, L)(sf^k(s), 2L)\\ &= (s, L)^{j-3}(sf^k(s)f^{2k}(s), 3L) = (s, L)^{j-3}(sf^k(s)f^k(s), 3L)\\ &= (s, L)^{j-3}(sf^k(s^2), 3L) = \cdots = (sf^k(s^{j-1}), jL).\end{aligned}$$

[25] For all finite semigroups S, there is some π such that s^π is idempotent for all $s \in S$. Here, we can choose $i = 2\pi$.

Since S is finite, there is some $i \geq 1$ such that $s^{i\ell-1}s^{i\ell} = s^{2i\ell-1} = s^{i\ell-1}$ holds[25] and, for this i, we have

$$\begin{aligned}&\left(sf^k(s^{\ell-1}), \ell\right)\left(f^{iL-1}(s^\ell), \ell\right)\left(f^{iL-2}(s^\ell), \ell\right)\cdots\left(f^1(s^\ell), \ell\right)\\ &\quad= \left(sf^k(s^{\ell-1})f^{iL}(s^\ell), 2\ell\right)\left(f^{iL-2}(s^\ell), \ell\right)\cdots\left(f^1(s^\ell), \ell\right)\\ &\quad= \left(sf^k(s^{\ell-1})f^k(s^\ell), 2\ell\right)\left(f^{iL-2}(s^\ell), \ell\right)\cdots\left(f^1(s^\ell), \ell\right)\\ &\quad= \left(sf^k(s^{\ell-1}s^\ell), 2\ell\right)\left(f^{iL-2}(s^\ell), \ell\right)\cdots\left(f^1(s^\ell), \ell\right)\\ &\quad= \left(sf^k(s^{\ell-1}s^{2\ell}), 3\ell\right)\left(f^{iL-3}(s^\ell), \ell\right)\cdots\left(f^1(s^\ell), \ell\right)\\ &\quad= \cdots = \left(sf^k(s^{\ell-1}s^{(iL-1)\ell}), i\ell L\right) = \left(sf^k(s^{i\ell L-1}), i\ell L\right) = \left(sf^k(s^{i\ell-1}), i\ell L\right)\\ &\quad= (s, L)^{i\ell},\end{aligned}$$

where we have used $f^k = f^{iL}$ (which holds because L is a multiple of k) and $s^{i\ell L-1} = s^{i\ell-1}$ (which holds by the choice of i). Since all the iL many factors in the first line are from R and since (s, L) is also in R, we obtain $(s, L)^{i\ell} \in R^{i\ell} \cap R^{iL}$ and even $(s, L)^{i\ell} \cdot \Sigma^* \subseteq R^{i\ell} \cap R^{iL}$.

We conclude by showing that $(r, j) \in R^{i\ell} \cap R^{iL}$ implies $j \in \{i\ell L\} \cup I$. Once we have done this, we can consider the projection $\gamma : S \rtimes T \to T$ to the second component. It is a nearly injective homomorphism (since S is finite) and we (then) have $\gamma((s, L)^{i\ell} \cdot \Sigma^*) \subseteq \{i\ell L\} \cup I$. Therefore, we can apply Lemma 1.2.1.3 and obtain that $(s, L)^{i\ell}$ must have torsion, which is a contradiction.

Let $(r, j) \in R^{i\ell} \cap R^{iL}$. If j is in I, there is nothing to show. Therefore, let $j \notin I$. Since we have $(r, j) \in R^{i\ell}$, we can write

$$(r, j) = (r_{i\ell}, J_{i\ell}) \dots (r_1, J_1)$$

for $(r_1, J_1), \dots, (r_{i\ell}, J_{i\ell}) \in R$ and $J_1, \dots, J_{i\ell} \notin I$. The definition of L implies $J_1, \dots, J_{i\ell} \leq L$. Therefore, we have $j = J_{i\ell} + \dots + J_1 \leq i\ell L$. On the other hand, we can also write

$$(r, j) = (r'_{iL}, j_{iL}) \dots (r'_1, j_1)$$

for some $(r'_1, j_1), \dots, (r'_{iL}, j_{iL}) \in R$ because (r, j) is in R^{iL}. Again, we have $j_1, \dots, j_{iL} \notin I$ and, by the choice of ℓ, also $j_1, \dots, j_{iL} \geq \ell$. Thus, we have $j = j_{iL} + \dots + j_1 \geq iL\ell$ and, therefore, $j = i\ell L$. □

Remark 1.2.1.5. That T needs to be an infinite subsemigroup in Theorem 1.2.1.4 is not very restrictive. The only other subsemigroups of an ideal small extension $\hat{T}$ of q^+ are contained in the finite ideal $\hat{T} - q^+$ and, thus, entirely unrelated to the monogenic free semigroup q^+.

Of course, it would be interesting to find further examples of non-automaton semigroups. One candidate for which the above methods might still be applicable is the semigroup $q^+ \cup p^+ \cup \{\mathbb{0}\}$, in which $\mathbb{0}$ is a zero and we have $qp = pq = \mathbb{0}$.

Open Problem 1.2.1.6. Is $q^+ \cup p^+ \cup \{\mathbb{0}\}$ an automaton semigroup?

1.3 Inverse Automaton Semigroups

In this section, we will study inverse automaton semigroups more closely. They form a natural intermediate step between automaton semigroups and automaton groups and have, in fact, turned out to be useful in tackling the complexity of the word problem for automaton groups (which we will discuss in more detail in Section 2.1).

We begin by giving a non-trivial example of an inverse automaton monoid: we present the free inverse monoid in one generator as an automaton-inverse monoid. This fits in the line of our previous examples as we have already seen that the free monoid on one generator is an automaton monoid and that the free group in one generator is an automaton group (see Example 0.2.3.5 for both). Additionally, we have discussed that the free semigroup in one generator is not an automaton semigroup in Section 1.2.

After looking at the free inverse monoid in one generator, we will see that any inverse automaton semigroup (or monoid) is already generated by an $\overline{\mathscr{S}}$-automaton, i. e. that it is an automaton-inverse semigroup (or monoid). To show this, we will exploit the close connection between inverse semigroups and partial one-to-one mappings expressed

in the so-called Preston-Vagner theorem (see e. g. [How95, Theorem 5.1.7, p. 150] or [Pet84, p. 168]). In its classical formulation, it states that every inverse semigroup S is a subsemigroup of the symmetric inverse semigroup $\mathscr{I}(S)$. In this way, it is the inverse semigroup counterpart to Cayley's theorem for groups, which states that every group G is a subgroup of the symmetric group over G. We will first prove a generalized variant of the Preston-Vagner theorem for (possibly non-faithful) semigroup actions. Then, we will use this generalization to construct an $\overline{\mathscr{S}}$-automaton generating the same semigroup as some given $\mathscr{S}$-automaton if the semigroup is inverse.

Using the same construction, we can also extend a result by Cain stating that every complete automaton semigroup that happens to be a group is already an automaton group [Cai09, Proposition 3.1] to (partial) automaton semigroups: if a group is a (partial) automaton semigroup, it is an automaton group.

Attribution. The idea to modify the adding machine to generate the free inverse monoid in one generator is inspired by a construction due to Olijnyk, Sushchansky and Słupik [OSS10, Fig. 8] which presents the free inverse semigroup in one generator as a subsemigroup of an automaton monoid. The actual modification is joint work with Daniele D'Angeli and Emanuele Rodaro [3, Example 2.3], [9, Example 23].

The result on the equivalence of inverse automaton semigroups and automaton-inverse semigroups is also joint work with Daniele D'Angeli and Emanuele Rodaro [9]; the monoid case is a direct extension of this. The idea is based on the similar approach used by Cain for groups [Cai09, Proposition 3.1] in the setting of complete automaton semigroups. The Preston-Vagner theorem is a classical result for inverse semigroups. The theorem itself and references to the original works by Preston and by Vagner can be found for example in [How95, Theorem 5.1.7, p. 150] or [Pet84, p. 168]. The proof of the generalized variant presented here (more or less loosely) follows the proof given by Howie [How95, Theorem 5.1.7, p. 150] for the classical variant.

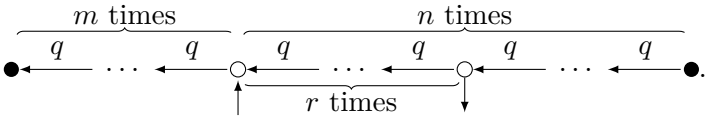
In general, inverse automaton semigroups seem to be much less studied than automaton groups or even (complete) automaton semigroups. There is the already mentioned work by Olijnyk, Sushchansky and Słupik [OSS10] investigating partial one-to-one functions defined by automata, which gives further references to other works by (some of) these authors. Other works mentioning self-similar inverse semigroups are due to Nekrashevych (e. g. [Nek06]).

1.3.1 The Monogenic Free Inverse Monoid

In this subsection, we are going to present the monogenic free inverse monoid as an inverse automaton monoid. We start, however, with discussing a graphical way of presenting the elements of free inverse monoids.

Munn Trees for the Monogenic Free Inverse Monoid. A good way to understand free inverse semigroups and monoids is to use their Munn tree presentation (see [Mun74] for Munn's original work or, for example, [How95, Example 5.10.7]). We will only give a

semi-formal description of Munn trees in the case of the monogenic free inverse monoid in the generator q. Here, a *Munn tree* can graphically be depicted as



It is a non-empty, finite, (weakly) connected subgraph of the bi-infinite directed line graph whose edges are labeled by q and has two dedicated nodes: the *initial* node (marked with an entering arrow) and the *final* node (marked with a leaving arrow). Thus, it can also be characterized by a triple m, n, r of natural numbers with $-m \leq r \leq n$. The number m is the length of the graph to the left of the initial node, n is the length to the right and r is the distance of the final node from the initial node (negative values indicate a position to the left, positive values a position to the right).

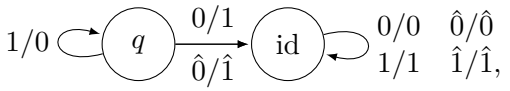
The elements of the monogenic free inverse monoid are in bijection with such Munn trees [Mun74, Theorem 2.8]. The bijection is given by taking the labeling of an arbitrary path in the Munn tree which starts in the initial node, ends in the final node and visits each node at least once. Here, a directed edge can also be used in the opposite direction by using the inverse of its label. For the above Munn tree, we can, for example, take the path labeled by $q^m\bar{q}^{m+n}q^{n-r}$. Thus, if we show that all elements $q^m\bar{q}^{m+n}q^{n-r}$ with $-m \leq r \leq n$ of a monogenic inverse monoid are pairwise distinct, we have shown that the monoid is a free inverse monoid. Because mapping an element to its inverse is a bijection on an inverse monoid,[26] we can alternatively show that all elements $\bar{q}^{n-r}q^{m+n}\bar{q}^m$ with $-m \leq r \leq n$ are pairwise distinct. Since this is the approach we are eventually going to use, we state this result in the following fact.

[26] In fact, it is an anti-automorphism.

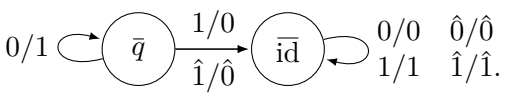
Fact 1.3.1.1. *A monogenic inverse monoid in the generator q is free if (and only if) all elements $\bar{q}^{n-r}q^{m+n}\bar{q}^m$ with $-m \leq r \leq n$ are pairwise disjoint.*

After these preparations, we are now ready to give an automaton generating the monogenic free inverse monoid as an automaton-inverse monoid.

Example 1.3.1.2 ([9, Example 23], [3, Example 2.3]; compare to [OSS10, Fig. 8])**.** Let us extend the adding machine from Example 0.2.1.4 into the $\overline{\mathscr{S}}$-automaton $\mathcal{T}$



whose inverse automaton is



We claim that the inverse monoid $M = \overline{\mathscr{M}}(\mathcal{T})$ generated by $\mathcal{T}$ is the monogenic free inverse monoid. By definition, it is an inverse monoid and, because we have $\mathrm{id} = \overline{\mathrm{id}} = \varepsilon$ in M, the state q is in M the single generator (as an inverse monoid). Thus, by Fact 1.3.1.1, we only have to show that $\bar{q}^{n-r}q^{m+n}\bar{q}^{m}$ with $-m \leq r \leq n$ is distinct to $\bar{q}^{n'-r'}q^{m'+n'}\bar{q}^{m'}$ with $-m' \leq r' \leq n'$ in M whenever we have $m \neq m'$, $r \neq r'$ or $n \neq n'$.

In order to give a witness on which the two state sequences act differently, we extend the notation $\partial \operatorname{bin} i$ from Example 0.2.1.4: for a natural number i with $0 \leq i < 2^\ell$, let $\partial \operatorname{bin}_\ell i$ denote the reverse/least significant bit first binary representation of i with length $\ell \geq 1$ over the alphabet $\{0, 1\}$. To obtain an actual length of ℓ, we possibly have to add trailing zeros. For an integer z outside the range $0 \leq z < 2^\ell$, we define $\partial \operatorname{bin} z$ as $\partial \operatorname{bin} i$ where i is the smallest non-negative representative of the congruence class of z modulo 2^ℓ (i. e. we have $0 \leq i < 2^\ell$ and that z and i are congruent modulo 2^ℓ). In particular, we have $\partial \operatorname{bin}_\ell 0 = 0^\ell$ and $\partial \operatorname{bin}_\ell(-1) = 1^\ell$. We have already seen in Example 0.2.1.4 that q acts as an increment on $\partial \operatorname{bin} i$ and $\bar{q}$ acts as a decrement. In the same way, we have $q^j \circ \partial \operatorname{bin}_\ell i = \partial \operatorname{bin}_\ell(i + j)$ and $\bar{q}^j \circ \partial \operatorname{bin}_\ell i = \partial \operatorname{bin}_\ell(i - j)$.

Now, suppose we have $m \neq m'$, $r \neq r'$ or $n \neq n'$ for $-m \leq r \leq n$ and $-m' \leq r' \leq n'$. We need to show that $\boldsymbol{p} = \bar{q}^{n-r}q^{m+n}\bar{q}^{m}$ and $\boldsymbol{p'} = \bar{q}^{n'-r'}q^{m'+n'}\bar{q}^{m'}$ are different in M.

First, suppose that we have $r = -(n - r) + m + n - m \neq -(n' - r') + m' + n' - m' = r'$. We choose ℓ large enough so that r and r' are distinct modulo 2^ℓ. Then, we have

$$\boldsymbol{p} \circ \partial \operatorname{bin}_\ell 0 = \partial \operatorname{bin}_\ell(-(n - r) + m + n - m + 0) = \partial \operatorname{bin}_\ell r \neq \partial \operatorname{bin}_\ell r' = \boldsymbol{p'} \circ \partial \operatorname{bin}_\ell 0.$$

Next, suppose that we have ($r = r'$ but) $m \neq m'$. Without loss of generality, we only consider the case $m < m'$. To avoid overflows in the following calculation, we choose ℓ large enough with $m + n < 2^\ell$. In particular, this also ensures $m + r \leq m + n < 2^\ell$ and we have

$$\begin{aligned}\bar{q}^{n-r}q^{m+n}\bar{q}^{m} \circ (\partial \operatorname{bin}_\ell m)\,\hat{0} = \bar{q}^{n-r}q^{m+n} \circ (\partial \operatorname{bin}_\ell 0)\,\hat{0} = \bar{q}^{n-r} \circ (\partial \operatorname{bin}_\ell(m + n))\,\hat{0} \\ = (\partial \operatorname{bin}_\ell(m + r))\,\hat{0}\end{aligned}$$

since we have always already entered id or $\overline{\mathrm{id}}$ before reading $\hat{0}$. On the other hand, we have that $\boldsymbol{p'} \circ (\partial \operatorname{bin}_\ell m)\,\hat{0}$ is undefined because, for $d = m' - m > 0$, already $\bar{q}^{m'} \circ (\partial \operatorname{bin}_\ell m)\,\hat{0} = \bar{q}^{m+d} \circ (\partial \operatorname{bin}_\ell m)\,\hat{0} = \bar{q}^{d} \circ (\partial \operatorname{bin}_\ell 0)\,\hat{0} = \bar{q}^{d} \circ 0^\ell\hat{0}$ is undefined.

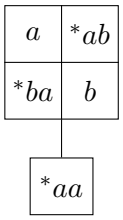
There remains the case ($r = r'$,) $m = m'$ but $n \neq n'$. Again, we may safely assume $n < n'$ (due to symmetry). We choose ℓ large enough so that $2^\ell \geq m + n + 1$ (i. e. we have $2^\ell - 1 - n + r \geq 2^\ell - 1 - n - m \geq 0$) and obtain

$$\begin{aligned}\bar{q}^{n-r}q^{m+n}\bar{q}^{m} \circ (\partial \operatorname{bin}_\ell(2^\ell - 1 - n))\,\hat{1} &= \bar{q}^{n-r}q^{m+n} \circ (\partial \operatorname{bin}_\ell(2^\ell - 1 - n - m))\,\hat{1} \\ &= \bar{q}^{n-r} \circ (\partial \operatorname{bin}_\ell(2^\ell - 1))\,\hat{1} \\ &= (\partial \operatorname{bin}_\ell(2^\ell - 1 - n + r))\,\hat{1}\end{aligned}$$

because we again always enter id or $\overline{\mathrm{id}}$ before reading $\hat{1}$. On the other hand, $\boldsymbol{p'} \circ (\partial \operatorname{bin}_\ell(2^\ell - 1 - n))\,\hat{1} = \bar{q}^{n'-r}q^{m+n'}\bar{q}^{m} \circ (\partial \operatorname{bin}_\ell(2^\ell - 1 - n))\,\hat{1}$ is undefined: we have already seen $q^{m+n}\bar{q}^{m} \circ (\partial \operatorname{bin}_\ell(2^\ell - 1 - n))\,\hat{1} = (\partial \operatorname{bin}_\ell(2^\ell - 1))\,\hat{1} = 1^\ell\,\hat{1}$ and the action of q on this word is undefined; thus, $q^{m+n'}\bar{q}^{m} \circ (\partial \operatorname{bin}_\ell(2^\ell - 1 - n))\,\hat{1}$ must be undefined.

	a	b	ab	ba	aa
a	aa	ab	aa	a	aa
b	ba	aa	b	aa	aa
ab	a	aa	ab	aa	aa
ba	aa	b	aa	ba	aa
aa	aa	aa	aa	aa	aa

(a) The multiplication table of B_2



(b) The eggbox presentation of B_2 (see e. g. [How95, p. 48] for an explanation)

Figure 1.2: The Brandt semigroup $B_2 = \{a, b, ab, ba, aa\}$

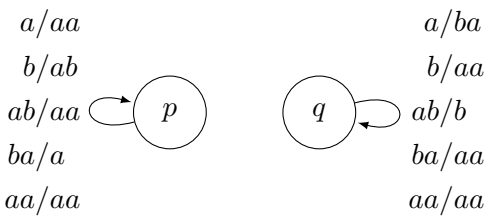
We have seen that the monogenic free monoid is an automaton semigroup in Example 0.2.3.5 while the monogenic free semigroup is not ([Cai09, Theorem 15] and Section 1.2 for the partial case). Now, in Example 1.3.1.2, we have seen that the monogenic free inverse monoid is an automaton semigroup. This raises the question whether the monogenic free inverse semigroup is an automaton semigroup.

Open Problem 1.3.1.3. Is the monogenic free inverse semigroup an automaton semigroup?

1.3.2 Inverse Automaton Semigroups and Automaton-Inverse Semigroups

An inverse automaton semigroup is an automaton semigroup that happens to be an inverse semigroup. On the other hand, an automaton-inverse semigroup is defined as the inverse semigroup generated by an $\overline{\mathscr{S}}$-automaton. To make this distinction a bit clearer, we will start by looking at a typical example for an inverse semigroup.

Example 1.3.2.1 (see also [9, Example 4])**.** The Brandt semigroup B_2 (see e. g. [How95, p. 32]) consists of the elements $\{a, b, ab, ba, aa\}$. From its multiplication table (or its eggbox presentation) given in Figure 1.2, it is easy to see that B_2 is an inverse semigroup (with $\bar{a} = b$ and $\bar{b} = a$).[27] The multiplication table also shows that B_2 acts faithfully on itself by left multiplication. We can use this fact to present B_2 as a complete automaton semigroup:[28] it is generated by the $\mathscr{S}$-automaton



over the alphabet $B_2 = \{a, b, ab, ba, aa\}$ (i. e. ab, ba and aa are considered as single letters). To better distinguish alphabet and state set, we have used p instead of a and q instead of b for the states.

[27] An alternative way of thinking about B_2 is to consider it as the syntactic semigroup of the language $(ab)^+$.

[28] In fact, this is the basic idea of how every finite semigroup can be presented as a complete automaton semigroup, see [Cai09, Proposition 4.6].

That B_2 can be generated by a (complete) $\mathscr{S}$-automaton proves that it is an inverse (complete) automaton semigroup. However, the automaton depicted above is not invertible and, therefore, not an $\overline{\mathscr{S}}$-automaton. To prove that B_2 is an automaton-inverse semigroup, we need an $\overline{\mathscr{S}}$-automaton $\mathcal{T}'$ with $\overline{\mathscr{S}}(\mathcal{T}') = B_2$.

Before we can go into detail about how to construct an $\overline{\mathscr{S}}$-automaton from an $\mathscr{S}$-automaton whose generated semigroup is inverse, we will first need some abstract, non-automaton-related results about inverse semigroups. We will start with a fact about partial actions of inverse semigroups.

Fact 1.3.2.2 (compare to [How95, Lemma 5.1.6])**.** *Let a semigroup S partially act from the left on some set X via $\alpha : S \to \mathscr{P}(X), s \mapsto \alpha_s$ and write $s \circ x$ for $\alpha_s(x)$ as well as $s \circ X$ for $\alpha_s(X) = \{\alpha_s(x) \mid x \in X, \alpha_s(x) \text{ defined}\}$. If S is an inverse semigroup, we have*

1. *$s\bar{s} \circ X = s \circ X$ and, thus, $\bar{s}s \circ X = \bar{s} \circ X$ for all $s \in S$ as well as*
2. *$s \circ X \cap \bar{t} \circ X = s\bar{s}\bar{t}t \circ X = \bar{t}ts\bar{s} \circ X$ for all $s, t \in S$.*

Proof. For the first statement, observe that the inclusion $s\bar{s} \circ X \subseteq s \circ X$ is trivial and the converse inclusion holds because of $s\bar{s} \circ X \supseteq s\bar{s}s \circ X = s \circ X$.

[29] See e. g. [How95, Theorem 5.1.1, p. 145]

The second equation of the second statement holds because $s\bar{s}$ and $\bar{t}t$ are idempotent and idempotents commute in inverse semigroups.[29] To prove the first equation, we use the first statement. This yields $s \circ X \cap \bar{t} \circ X = s\bar{s} \circ X \cap \bar{t}t \circ X$. We have $s\bar{s} \circ X \supseteq s\bar{s}\bar{t}t \circ X = \bar{t}ts\bar{s} \circ X \subseteq \bar{t}t \circ X$, which shows the inclusion $s \circ X \cap \bar{t} \circ X \supseteq s\bar{s}\bar{t}t \circ X$. For the other inclusion, let $x \in s\bar{s} \circ X \cap \bar{t}t \circ X$, i. e. there are $y, z \in X$ with $x = s\bar{s} \circ y = \bar{t}t \circ z$ (in particular, both defined). We have $s\bar{s}\bar{t}t \circ x = s\bar{s}\bar{t}t\bar{t}t \circ z = s\bar{s}\bar{t}t \circ z = s\bar{s} \circ x = s\bar{s}s\bar{s} \circ y = s\bar{s} \circ y = x$ (all defined), which shows that x is in $s\bar{s}\bar{t}t \circ X$. □

Next, we will prove the following generalization of the Preston-Vagner theorem (see e. g. [How95, Theorem 5.1.7, p. 150] or [Pet84, p. 168] for the classical formulation of the Preston-Vagner theorem; also for the original references), which states that an inverse semigroup that partially (and faithfully) acts on a set as a semigroup already partially (and faithfully) acts on this set as an inverse semigroup (i. e. using partial one-to-one mappings). While this result is interesting on its own, it will also be the key ingredient for our automaton construction later on.

Proposition 1.3.2.3 ([9, Lemma 24])**.** *Let a semigroup S partially act from the left on some set X via $\alpha : S \to \mathscr{P}(X), s \mapsto \alpha_s$ and write $s \circ x$ for $\alpha_s(x)$ as well as $s \circ X$ for $\alpha_s(X) = \{\alpha_s(x) \mid x \in X, \alpha_s(x) \text{ defined}\}$. Additionally, let S be an inverse semigroup.*

Then, the restriction

$$\begin{aligned} \varphi_s : \bar{s} \circ X &\to s \circ X \\ \bar{s} \circ x &\mapsto s\bar{s} \circ x \end{aligned}$$

of the partial action of s is total, one-to-one and surjective for all $s \in S$.

Furthermore, if we consider φ_s as a partial function $X \rightharpoonup X$, then the map

$$\begin{aligned} \varphi : S &\to \mathscr{I}(X) \\ s &\mapsto \varphi_s \end{aligned}$$

is a homomorphism of inverse semigroups and it is injective if the partial action α is faithful.

Proof (following the outline of the proof of [How95, Theorem 5.1.7, p. 150]). First, we show that φ_s is total for all $s \in S$ as a function $\bar{s} \circ X \to s \circ X$. If $\bar{s} \circ x$ is defined for some $x \in X$, we have that $\bar{s}s\bar{s} \circ x = \bar{s} \circ x$ must also be defined. This implies, in particular, that $s\bar{s} \circ x$ must be defined.

Next, we show that φ_s is one-to-one. Suppose, we have $s\bar{s} \circ x = s\bar{s} \circ y$ (and that both are defined). This implies $\bar{s} \circ x = \bar{s}s\bar{s} \circ x = \bar{s}s\bar{s} \circ y = \bar{s} \circ y$ (where all terms are defined).

By Fact 1.3.2.2, we have $s \circ X = s\bar{s} \circ X$ for all $s \in S$. This, in particular, implies $\operatorname{im} \varphi_s = s\bar{s} \circ X = s \circ X$, i. e. that φ_s is surjective (as a map $\bar{s} \circ X \to s \circ X$).

If we consider φ_s as a partial function $X \rightharpoonup X$ (i. e. as an element of $\mathscr{I}(X)$), then its semigroup inverse $\overline{\varphi_s}$ is $\varphi_{\bar{s}}$. To see this, observe that we have $\operatorname{dom} \overline{\varphi_s} = \operatorname{im} \varphi_s = s \circ X = \bar{\bar{s}} \circ X = \operatorname{dom} \varphi_{\bar{s}}$ and $\varphi_{\bar{s}}(\varphi_s(\bar{s} \circ x)) = \bar{s}s\bar{s} \circ x = \bar{s} \circ x$ for all $x \in X$ such that $\bar{s} \circ x$ is defined.

We use this to show that φ is a homomorphism, i. e. that $\varphi_s \circ \varphi_t = \varphi_{st}$. We have $\operatorname{dom} \varphi_t \circ \varphi_s = \overline{\varphi_s}(\operatorname{im} \varphi_s \cap \operatorname{dom} \varphi_t) = \varphi_{\bar{s}}(s \circ X \cap \bar{t} \circ X) = \varphi_{\bar{s}}(s\bar{s}\bar{t} \circ X)$ where we have used Fact 1.3.2.2 in the last equality. Since $s\bar{s}\bar{t} \circ X$ is a subset of $s \circ X = \operatorname{dom} \varphi_{\bar{s}}$, we have $\varphi_{\bar{s}}(s\bar{s}\bar{t} \circ X) = \bar{s}s\bar{s}\bar{t} \circ X = \bar{s}\bar{t} \circ X = \bar{s}\bar{t} \circ X = \overline{ts} \circ X$ where we have again used Fact 1.3.2.2 in the second last equality. Summarizing this, we have $\operatorname{dom} \varphi_t \circ \varphi_s = \overline{ts} \circ X = \operatorname{dom} \varphi_{ts}$. This shows $\varphi_s \circ \varphi_t = \varphi_{st}$ since the values of $\varphi_t \circ \varphi_s$ and φ_{ts} coincide on all elements from this domain by definition.

It remains to show that φ is injective if the partial action α is faithful, i. e. that $\varphi_s = \varphi_t$ implies $s = t$ for all $s, t \in S$. Thus, assume $\varphi_s = \varphi_t$, which also implies $\varphi_{\bar{s}} = \overline{\varphi_s} = \overline{\varphi_t} = \varphi_{\bar{t}}$. Additionally, we have $\bar{s} \circ X = \operatorname{dom} \varphi_s = \operatorname{dom} \varphi_t = \bar{t} \circ X$ and, symmetrically, $s \circ X = t \circ X$. In particular, we have $\varphi_s(\bar{t} \circ y) = s\bar{t} \circ y$ (or both undefined) for all $y \in X$.

This allows us to show that $\bar{t}s$ is idempotent in this case. We have to show $\bar{t}s \circ x = \bar{t}s\bar{t}s \circ x$ (or both undefined) for all $x \in X$. If $\bar{t}s \circ x$ is undefined, then $\bar{t}s\bar{t}s \circ x$ must also be undefined. If $\bar{t}s \circ x$ is defined, we have have $\bar{t}s \circ x = \bar{t}t\bar{t}s \circ x = \bar{t} \circ \varphi_t(\bar{t}s \circ x) = \bar{t} \circ \varphi_s(\bar{t}s \circ x) = \bar{t}s\bar{t}s \circ x$.

To finally show $s = t$, we show $s \circ x = t \circ x$ (or both undefined) for all $x \in X$ and then use the assumption that α is faithful. For simplicity, assume that $s \circ x$ is defined for some $x \in X$. Then, we have

$$s \circ x = s\bar{s}s \circ x = s \circ \varphi_{\bar{s}}(s \circ x) = s \circ \varphi_{\bar{t}}(s \circ x) = s\bar{t}s \circ x = s\bar{t}t\bar{t}s \circ x = s\bar{t}s\bar{t}t \circ x$$

(and that all terms are defined) where we have used that $\bar{t}t$ and $\bar{t}s$ are idempotent in the last step. Because φ_s is defined on $\bar{t} \circ X = \bar{s} \circ X$, we can continue with

$$\begin{aligned} &= s\bar{t} \circ \varphi_s(\bar{t}t \circ x) = s\bar{t} \circ \varphi_t(\bar{t}t \circ x) = s\bar{t}t\bar{t}t \circ x = s\bar{t}t \circ x = \varphi_s(\bar{t}t \circ x) = \varphi_t(\bar{t}t \circ x) \\ &= t\bar{t}t \circ x = t \circ x \end{aligned}$$

(where all terms are defined). If $t \circ x$ is defined, we can use the same calculation backwards to show $t \circ x = s \circ x$. □

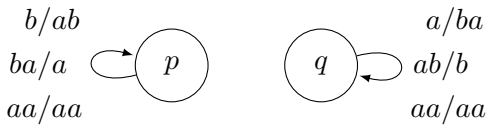
We will now return from abstract inverse semigroups to automaton semigroups. First, we continue our example from above.

Example 1.3.2.4 (see also [9, Example 4])**.** We can use Proposition 1.3.2.3 to restrict the faithful action of B_2 on itself from Example 1.3.2.1 to a faithful partial action using one-to-one partial mappings.[30] We have $\bar{a}B_2 = bB_2 = \{b, ba, aa\}$ and $\bar{b}B_2 = aB_2 = \{a, ab, aa\}$. We can use

$$\begin{aligned} a' : bB_2 &\to aB_2 \\ b &\mapsto ab \\ ba &\mapsto a \\ aa &\mapsto aa \end{aligned} \qquad \text{and} \qquad \begin{aligned} b' : aB_2 &\to bB_2 \\ a &\mapsto ba \\ ab &\mapsto b \\ aa &\mapsto aa \end{aligned}$$

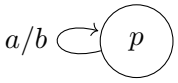
as the partial action of a and b on B_2, respectively.

This leads to the $\overline{\mathscr{S}}$-automaton



generating B_2 (where we have again used p and q instead of a and b for the state set and where ab, ba and aa have to be considered as single letters). Indeed, this automaton demonstrates the basic idea of constructing an $\overline{\mathscr{S}}$-automaton from an $\mathscr{S}$-automaton that generates the same (inverse) semigroup.

Note, however, that, at least in this special case, we can find a smaller $\overline{\mathscr{S}}$-automaton generating B_2 (as an inverse semigroup). First, the state q acts in the same way as the state $\bar{p}$ of the inverse automaton. Thus, the whole state can be dropped without altering the generated inverse semigroup. Second, the letter aa does not seem to be very useful. In fact, the very simple $\overline{\mathscr{S}}$-automaton



over the alphabet $\Sigma = \{a, b\}$ generates B_2 as an inverse semigroup. The partial action of the state p is to map a^n to b^n for all $n \in \mathbb{N}$; this corresponds to a in B_2. The partial action of its inverse $\bar{p}$ obviously maps b^n to a^n for all $n \in \mathbb{N}$ and corresponds to b in B_2. The partial action of p^2 is the partial function that is undefined everywhere (except on the empty word) and p^2, thus, corresponds to the zero aa in B_2. The other two partial functions in the automaton semigroup (under the functional view) are given by $a^n \mapsto a^n$ and by $b^n \mapsto b^n$ for all $n \in \mathbb{N}$ and correspond to ba and ab in B_2.

The idea from the example also works in the general case. If we have an $\mathscr{S}$-automaton $\mathcal{T}$ such that its generated semigroup $\mathscr{S}(\mathcal{T})$ is an inverse semigroup, then we can construct an $\overline{\mathscr{S}}$-automaton $\mathcal{T}'$ as a subautomaton of $\mathcal{T}$ with $\mathscr{S}(\mathcal{T}')$ isomorphic to $\mathscr{S}(\mathcal{T})$. Before we describe the construction, however, we first show that the isomorphism between $\mathscr{S}(\mathcal{T})$ and $\mathscr{S}(\mathcal{T}')$ also implies that $\overline{\mathscr{S}}(\mathcal{T}')$ and $\mathscr{S}(\mathcal{T})$ are isomorphic.

[30] In fact, for the example, the classical formulation of the Preston-Vagner theorem mentioned above suffices.

Fact 1.3.2.5. *Let $\mathcal{T}$ be an $\overline{\mathscr{S}}$-automaton. Then, we have*

$$\mathscr{S}(\mathcal{T}) \text{ is an inverse semigroup} \implies \mathscr{S}(\mathcal{T}) \text{ is (isomorphic to) } \overline{\mathscr{S}}(\mathcal{T}) \text{ and}$$
$$\mathscr{M}(\mathcal{T}) \text{ is an inverse monoid} \implies \mathscr{M}(\mathcal{T}) \text{ is (isomorphic to) } \overline{\mathscr{M}}(\mathcal{T}).$$

Proof. Let $\mathcal{T} = (Q, \Sigma, \delta)$ and consider the map

$$\iota : \mathscr{M}(\mathcal{T}) = Q^*/{=_{\mathcal{T}}} \to \widetilde{Q}^*/{=_{\widetilde{\mathcal{T}}}} = \overline{\mathscr{M}}(\mathcal{T})$$
$$[\boldsymbol{q}]_{\mathcal{T}} \mapsto [\boldsymbol{q}]_{\widetilde{\mathcal{T}}}.$$

It is clearly a well-defined, injective, homomorphism of monoids. To show that it is also surjective, we show that, for every $\bar{q} \in \overline{Q}$ (that is not the neutral element in $\mathscr{M}(\mathcal{T})$), there is some $\hat{\boldsymbol{q}} \in Q^+$ with $\hat{\boldsymbol{q}} =_{\widetilde{\mathcal{T}}} \bar{q}$. Since $\mathscr{M}(\mathcal{T})$ is an inverse monoid, there must be some $\hat{\boldsymbol{q}} \in Q^+$ with $q\hat{\boldsymbol{q}}q =_{\mathcal{T}} q$ and $\hat{\boldsymbol{q}}q\hat{\boldsymbol{q}} =_{\mathcal{T}} \hat{\boldsymbol{q}}$. This implies $q\hat{\boldsymbol{q}}q =_{\widetilde{\mathcal{T}}} q$ and $\hat{\boldsymbol{q}}q\hat{\boldsymbol{q}} =_{\widetilde{\mathcal{T}}} \hat{\boldsymbol{q}}$ or, in other words, that $\hat{\boldsymbol{q}}$ is a semigroup inverse to q in $\overline{\mathscr{M}}(\mathcal{T})$. Since $\bar{q}$ is also a semigroup inverse of q in $\overline{\mathscr{M}}(\mathcal{T})$ and since this monoid is inverse, we obtain $\bar{q} =_{\widetilde{\mathcal{T}}} \hat{\boldsymbol{q}}$. The statement about semigroups follows by using the appropriate restriction of ι. □

Theorem 1.3.2.6 (extension of [9, Theorem 25] to monoids)**.** *A semigroup is an inverse automaton semigroup if and only if it is an automaton-inverse semigroup and a monoid is an inverse automaton monoid if and only if it is an automaton-inverse monoid.*

Proof. If S is an automaton-inverse semigroup, there is some $\overline{\mathscr{S}}$-automaton $\mathcal{T}$ with $\overline{\mathscr{S}}(\mathcal{T}) = S$. Thus, S is an inverse semigroup and it is generated as an automaton semigroup by the $\mathscr{S}$-automaton $\mathcal{T} \uplus \overline{\mathcal{T}}$. The monoid statement follows in the same way.

For the other direction, suppose that $S = \mathscr{S}(\mathcal{T})$ for some (not necessarily invertible) $\mathscr{S}$-automaton $\mathcal{T} = (Q, \Sigma, \delta)$ is an inverse semigroup. In particular, for every $p \in Q$, there is some $\overline{\boldsymbol{p}} \in Q^+$ such that $\overline{\boldsymbol{p}}$ is the semigroup inverse of p in S. We need to find some $\overline{\mathscr{S}}$-automaton $\mathcal{T}'$ such that $\overline{\mathscr{S}}(\mathcal{T}')$ is (isomorphic to) S. In fact, we will give an $\overline{\mathscr{S}}$-automaton $\mathcal{T}'$ such that $\mathscr{S}(\mathcal{T}')$ is already (isomorphic to) S; this is a stronger statement by Fact 1.3.2.5.

We use $\mathcal{T}' = (Q', \Sigma, \delta')$ as this automaton where $Q' = \{q' \mid q \in Q\}$ is a disjoint copy of Q and the transitions are given by

$$\delta' = \{p' \xrightarrow{a/b} q' \mid p \xrightarrow{a/b} q \in \delta, a \in \overline{\boldsymbol{p}} \circ \Sigma\}.$$

Here, $\overline{\boldsymbol{p}} \circ \Sigma$ is $\overline{\boldsymbol{p}} \circ \Sigma = \{\overline{\boldsymbol{p}} \circ a \mid a \in \Sigma, \overline{\boldsymbol{p}} \circ a \text{ defined}\}$. The automaton $\mathcal{T}'$ is a (disjoint copy of a) subautomaton of $\mathcal{T}$ and, thus, an $\mathscr{S}$-automaton. Furthermore, if we restrict the partial action $p' \circ$ of $p' \in Q'$ to a partial function $\Sigma \rightharpoonup \Sigma$, then this coincides with the restriction of the action $p \circ$ of $p \in Q$ to a partial function $\overline{\boldsymbol{p}} \circ \Sigma \rightharpoonup \Sigma$ by construction. By Proposition 1.3.2.3, the latter is one-to-one, which means that $\mathcal{T}'$ must be invertible and, thus, an $\overline{\mathscr{S}}$-automaton.

We claim that the partial action $p' \circ$ of $p' \in Q'$ is, in fact, the restriction of $p \circ$ to a partial function $\overline{\boldsymbol{p}} \circ \Sigma^* \rightharpoonup \Sigma^*$ (where $\overline{\boldsymbol{p}} \circ \Sigma^*$ is $\overline{\boldsymbol{p}} \circ \Sigma^* = \{\overline{\boldsymbol{p}} \circ u \mid u \in \Sigma^*, \overline{\boldsymbol{p}} \circ u \text{ defined}\}$). If this is true, then we have that $\mathscr{S}(\mathcal{T}')$ is (isomorphic to) $S = \mathscr{S}(\mathcal{T})$ by Proposition 1.3.2.3.

Clearly, every run in $\mathcal{T}'$ yields a corresponding run in $\mathcal{T}$. Thus, if $p' \circ u$ is defined for some $u \in \Sigma^*$, then we have $p' \circ u = p \circ u$ (which must be defined, in particular). Therefore, to show the semigroup statement, we only have to show that the domain of $p' \circ$ is $\overline{\boldsymbol{p}} \circ \Sigma^*$. To do this, we let $D_p = \overline{\boldsymbol{p}} \circ \Sigma^*$ and $D'_p = \operatorname{dom} p' \circ \subseteq \operatorname{dom} p \circ$ for all $p \in Q$ and show

$$D_p \cap \Sigma^n = D'_p \cap \Sigma^n \qquad (\bigstar)$$

for all $n \in \mathbb{N}$ by induction.

For $n = 0$, we have $D_p \cap \Sigma^0 = \{\varepsilon\} = D'_p \cap \Sigma^0$. For $n > 0$, we first observe that we have $D_p \cap \Sigma = D'_p \cap \Sigma$ by construction. Suppose we have $a \in D_p \cap \Sigma = D'_p \cap \Sigma$ for some $a \in \Sigma$. Then, we have $p' \circ a = b = p \circ a$ (all defined) for some $b \in \Sigma$ and $p' \cdot a = q'$ for some $q' \in Q'$. In $\mathcal{T}$, this yields the cross diagram

$$\begin{array}{ccc} & a & \\ p & \dashrightarrow & q \\ & b & \\ \overline{\boldsymbol{p}} & \dashrightarrow & \hat{\boldsymbol{q}} \\ & c & \\ p & \dashrightarrow & r \\ & b & \end{array}$$

for some $c \in \Sigma$, $r \in Q$ and $\hat{\boldsymbol{q}} \in Q^+$. In particular, we have $c \in \overline{\boldsymbol{p}} \circ \Sigma$ and, thus, that $p' \circ c$ is defined and equal to b. Since $\mathcal{T}'$ is invertible, we obtain $c = a$ and $r = p \cdot c = p \cdot a = q$. This means that we have the cross diagram

$$\begin{array}{ccc} & b & \\ \overline{\boldsymbol{p}} & \dashrightarrow & \hat{\boldsymbol{q}} \\ & a & \end{array} \quad \text{in } \mathcal{T}.$$

Since we will need it later on to apply the induction hypothesis, we will continue by showing that $\hat{\boldsymbol{q}} = \overline{\boldsymbol{q}}$ in S holds. We do this by showing that $\hat{\boldsymbol{q}}$ is a semigroup inverse to q in S. Since, in an inverse semigroup, the inverse is unique, the equality then follows.

Let $u \in \Sigma^*$ be arbitrary. If $q \circ u$ is undefined, then $q\hat{\boldsymbol{q}}q \circ u$ is also undefined. If it is defined, we have the cross diagram

$$\begin{array}{ccccc} & a & & u & \\ p & \dashrightarrow & q & \dashrightarrow & \\ & b & & v & \\ \overline{\boldsymbol{p}} & \dashrightarrow & \hat{\boldsymbol{q}} & \dashrightarrow & \\ & a & & w & \\ p & \dashrightarrow & q & \dashrightarrow & \\ & b & & v & \end{array}$$

in $\mathcal{T}$ for $v = q \circ u$ and some $w \in \Sigma^*$ because $\overline{\boldsymbol{p}}$ is the inverse of p in S. Thus, we have $q \circ u = v = q\hat{\boldsymbol{q}}q \circ u$ (all defined).

On the other hand, if we now let $v \in \Sigma^*$ be arbitrary, then we have that $\hat{\boldsymbol{q}}q\hat{\boldsymbol{q}} \circ v$ is undefined if $\hat{\boldsymbol{q}} \circ v$ is. If it is defined, however, we obtain the cross diagram

$$\begin{array}{ccc} & b & v \\ \overline{\boldsymbol{p}} & \downarrow & \hat{\boldsymbol{q}} \ \downarrow \\ & a & u \\ p & \downarrow & q \ \downarrow \\ & b & w \\ \overline{\boldsymbol{p}} & \downarrow & \hat{\boldsymbol{q}} \ \downarrow \\ & a & u \end{array}$$

in $\mathcal{T}$ for $u = \hat{\boldsymbol{q}} \circ v$ and some (new) $w \in \Sigma^*$; again, because p is the inverse of $\overline{\boldsymbol{p}}$ in S. This shows $\hat{\boldsymbol{q}}q\hat{\boldsymbol{q}} \circ v = u = \hat{\boldsymbol{q}} \circ v$ (all defined) and, thus, that $\hat{\boldsymbol{q}}$ is indeed the inverse of q in S.

Now, for the inductive step, let $x' \in D_p \cap \Sigma^n = \overline{\boldsymbol{p}} \circ \Sigma^n$. We have to show $x' \in D'_p \cap \Sigma^n$, which is implied by $x' \in D'_p$. We can write $x' = ax$ for some $x \in \Sigma^{n-1}$ without loss of generality since a was chosen arbitrarily from $D_p \cap \Sigma = D'_p \cap \Sigma$. Now, $ax \in D_p$ implies that there is some $c \in \Sigma$ and some $z \in \Sigma^{n-1}$ with $ax = \overline{\boldsymbol{p}} \circ cz$. Thus, we have the cross diagram

$$\begin{array}{ccc} & c & z \\ \overline{\boldsymbol{p}} & \downarrow & \hat{\boldsymbol{r}} \ \downarrow \\ & a & x \\ p & \downarrow & q \ \downarrow \\ & b & y \\ \overline{\boldsymbol{p}} & \downarrow & \hat{\boldsymbol{q}} \ \downarrow \\ & a & x \end{array}$$

in $\mathcal{T}$ for some $y \in \Sigma^{n-1}$ and some $\hat{\boldsymbol{r}} \in Q^+$. The highlighted part implies $x = \hat{\boldsymbol{q}} \circ y \in \hat{\boldsymbol{q}} \circ \Sigma^{n-1} = \overline{\boldsymbol{q}} \circ \Sigma^{n-1} = D_q \cap \Sigma^{n-1} = D'_q \cap \Sigma^{n-1}$ by induction. This means that $q' \circ x$ is defined and that we, thus, have the cross diagram

$$\begin{array}{ccc} & a & x \\ p' & \downarrow & q' \ \downarrow \\ & b & y \end{array}$$

in $\mathcal{T}'$, which shows $ax \in D'_p$.

For the other direction, let $x' \in D'_p \cap \Sigma^n$. We have to show $x' \in D_p \cap \Sigma^n$, which is implied by $x' \in D_p$. Again, we can, without loss of generality, write $x' = ax$ for some $x \in \Sigma^{n-1}$. Since ax is in D'_p, we have that $p' \circ ax$ is defined and, thus, that $p' \circ ax = p \circ ax = by$ for some $y \in \Sigma^{n-1}$. Together with the fact that p is the inverse of $\overline{\boldsymbol{p}}$ in S, this yields the cross diagram

$$\begin{array}{ccc} & a & x \\ p & \downarrow & q \ \downarrow \\ & b & y \\ \overline{\boldsymbol{p}} & \downarrow & \hat{\boldsymbol{q}} \ \downarrow \\ & a & z \\ p & \downarrow & q \ \downarrow \\ & b & y \end{array}$$

in $\mathcal{T}$ for $z = \hat{\boldsymbol{q}} \circ y$. We have $z = \hat{\boldsymbol{q}} \circ y = \overline{\boldsymbol{q}} \circ y \in \overline{\boldsymbol{q}} \circ \Sigma^{n-1} = D_q \cap \Sigma^{n-1} = D'_q \cap \Sigma^{n-1}$ by induction, which implies the cross diagram

$$\begin{array}{ccc} & a & z \\ p' & \dashrightarrow & q' \dashrightarrow \\ & b & y \end{array}$$

in $\mathcal{T}'$. Since $\mathcal{T}'$ is invertible, we have that $q' \circ$ is one-to-one, which implies $x = z$ and, thus, together with the highlighted part of the above cross diagram, $ax \in D_p$.

We have shown (★) and, thus, the statement for semigroups. To extend it to monoids, we will show that $\boldsymbol{p} =_{\mathcal{T}} \varepsilon$ holds for $Q^+ \ni \boldsymbol{p} = p_\ell \dots p_2 p_1$ with $p_1, p_2, \dots, p_\ell \in Q$ if and only if $\boldsymbol{p}' =_{\mathcal{T}'} \varepsilon$ holds where $\boldsymbol{p}'$ is $\boldsymbol{p}' = p'_\ell \dots p'_2 p'_1$. Then, if there is some $\boldsymbol{p} \in Q^+$ with $\boldsymbol{p} =_{\mathcal{T}} \varepsilon$, we have the following chain of isomorphisms:

$$\mathscr{M}(\mathcal{T}) \simeq \mathscr{S}(\mathcal{T}) \simeq \mathscr{S}(\mathcal{T}') \simeq \overline{\mathscr{S}}(\mathcal{T}') \simeq \overline{\mathscr{M}}(\mathcal{T}')$$

where we have used Fact 1.3.2.5 in the third step. If there is no $\boldsymbol{p} \in Q^+$ with $\boldsymbol{p} =_{\mathcal{T}} \varepsilon$, then we have a similar chain of isomorphisms:

$$\mathscr{M}(\mathcal{T}) \simeq \mathscr{S}(\mathcal{T})^{\mathbb{1}} \simeq \mathscr{S}(\mathcal{T}')^{\mathbb{1}} \simeq \mathscr{M}(\mathcal{T}') \simeq \overline{\mathscr{M}}(\mathcal{T}')$$

where we have used Fact 1.3.2.5 again in the last step.

To show the actual claim, observe that $\boldsymbol{p}' =_{\mathcal{T}'} \varepsilon$ implies $\boldsymbol{p} =_{\mathcal{T}} \varepsilon$ because, in this case, the partial action $\boldsymbol{p}' \circ$ of $\boldsymbol{p}'$ is the identity mapping and, thus, defined on all of Σ^*. On the other hand, if we have $\boldsymbol{p} =_{\mathcal{T}} \varepsilon$, the inverse of $\boldsymbol{p}$ in $\mathscr{M}(\mathcal{T})$ must also be ε. Thus, by the above proof, $\boldsymbol{p}' \circ u$ is defined for all $u \in \varepsilon \circ \Sigma^* = \Sigma^*$ and we have $\boldsymbol{p}' \circ u = \boldsymbol{p} \circ u = u$ for all $u \in \Sigma^*$. □

The construction from the previous proof can also be used to show that an automaton semigroup that is a group is already an automaton group. This extends [Cai09, Proposition 3.1] to partial automata.

Corollary 1.3.2.7 ([Cai09, Proposition 3.1] for partial automata)**.** *A group is a (partial) automaton semigroup if and only if it is an automaton group.*

Proof. An automaton group $\mathscr{G}(\mathcal{T})$ is by definition the automaton semigroup $\mathscr{S}(\widetilde{\mathcal{T}})$.

For the other direction, suppose that $\mathscr{S}(\mathcal{T}) = G$ is a group for some $\mathscr{S}$-automaton $\mathcal{T} = (Q, \Sigma, \delta)$. Thus, there is some $\boldsymbol{e} \in Q^+$ such that $\boldsymbol{e}$ is the neutral element in G. Since the trivial group is certainly an automaton group, we may assume that G is not trivial, which implies that $\boldsymbol{e} \circ \Sigma = \{\boldsymbol{e} \circ a \mid a \in \Sigma, \boldsymbol{e} \circ a \text{ defined}\}$ is not empty. We will show that we have $\boldsymbol{q} \circ \Sigma = \boldsymbol{e} \circ \Sigma$ for all $\boldsymbol{q} \in Q^+$. To do this, we first observe that, for every $\boldsymbol{q} \in Q^+$, there is some $\boldsymbol{q}^{-1} \in Q^+$ such that $\boldsymbol{q}^{-1}$ is the (group) inverse of $\boldsymbol{q}$ in G. We have $\boldsymbol{q} \circ \Sigma = \boldsymbol{e}\boldsymbol{q} \circ \Sigma \subseteq \boldsymbol{e} \circ \Sigma$ and $\boldsymbol{e} \circ \Sigma = \boldsymbol{q}\boldsymbol{q}^{-1} \circ \Sigma \subseteq \boldsymbol{q} \circ \Sigma$.

Recall the construction of the $\overline{\mathscr{S}}$-automaton $\mathcal{T}'$ from $\mathcal{T}$ in the proof of Theorem 1.3.2.6. Applying this construction here, we obtain the automaton $\mathcal{T}' = (Q', \Sigma, \delta')$ where Q' is a disjoint copy of Q and the transitions are given by

$$\delta' = \{p' \xrightarrow{a/b} q' \mid p \xrightarrow{a/b} q \in \delta, a \in \boldsymbol{p}^{-1} \circ \Sigma = \boldsymbol{e} \circ \Sigma\}$$

and that $\overline{\mathscr{S}}(\mathcal{T}')$ is (isomorphic to) $\mathscr{S}(\mathcal{T})$. We claim that $\mathcal{T}'$ is a complete automaton over the alphabet $\boldsymbol{e} \circ \Sigma$, which implies that $\mathscr{S}(\mathcal{T})$ is (isomorphic to) the automaton group $\mathscr{G}(\mathcal{T}')$. Clearly, for every transition $p' \xrightarrow{a/b} q'$ in δ', we have $a \in \boldsymbol{p}^{-1} \circ \Sigma = \boldsymbol{e} \circ \Sigma$ and $b = p \circ b \in p \circ \Sigma = \boldsymbol{e} \circ \Sigma$. It remains to show that $p' \circ b$ is defined for all $b \in \boldsymbol{e} \circ \Sigma$ and this is the case if $p \circ b$ is defined for all $b \in \boldsymbol{e} \circ \Sigma$. So, let $b = \boldsymbol{e} \circ a$ (be defined) for some $a \in \Sigma$. Then, we have that $\boldsymbol{e} \circ a = \boldsymbol{p}^{-1} p \boldsymbol{e} \circ a$ is defined. This is only possible if also $p\boldsymbol{e} \circ a = p \circ \boldsymbol{e} \circ a = p \circ b$ is defined. □

1.4 Orbits of Automaton Semigroups

In this section, we will have a closer look at the orbits of automaton semigroups and groups. In the first subsection, we will show our main result that every infinite automaton semigroup (and, thus, also every infinite automaton group) admits an ω-word with an infinite orbit. This result is less obvious than it seems at first and solves a previously open problem [3, Open problem 4.3].[31] In fact, we will show a more general result stating that, if a suffix-closed language K over the state set of an $\mathscr{S}$-automaton has an infinite image in the generated semigroup, then it also admits an ω-word with an infinite K-orbit. We will apply this result to the whole semigroup, to finitely generated subsemigroups but also to principal left ideals of the semigroup (which form non-finitely generated subsemigroups in general). On the other hand, we will see that the analogue of this result does not hold for general non-finitely generated subsemigroups since it is not true for principal right or two-sided semigroup ideals.

[31] The problem was also communicated by Ievgen Bondarenko (although no reference exists).

While the main result has many interesting consequences (also for some algebraic decision problems that we will discuss in Subsection 2.3.3 and Subsection 2.4.3), its proof and the involved objects are also interesting on their own. The central idea is to extend the well-known result that an $\mathscr{S}$-automaton generates an infinite semigroup if and only if its dual does[32] to a generalized form of Schreier graphs for (automaton) semigroups. Here, we will prove a duality result, which will not only be central for the main result but can also be used to show some further interesting consequences. For example, it allows us to give a simple proof for a connection between the torsion of a semigroup element and the finiteness of its orbit under the action of the dual generalizing a result for automaton groups to semigroups, which we will give at the beginning of the second subsection. In connection with the existence of infinite torsion automaton groups, this result allows us to show that the ω-word with an infinite orbit can neither be assumed to be periodic nor to be ultimately periodic in general. On the other hand, we will also see from the dual connection that, if an automaton semigroup admits an ω-word with a finite orbit, then it already admits such an ω-word which is ultimately periodic or, in the case of a complete and reversible automaton, even periodic. Eventually, we will re-visit this short glimpse on (ultimately) periodic words with a finite orbit in Subsection 2.2.2 to show some undecidability result.

[32] See Corollary 1.4.1.12 (and the part before it) for references.

Attribution. The proof for the main result given in the first subsection has a rather convoluted history. The original question whether an infinite automaton semigroup always

admits a single ω-word with an infinite orbit was put on record in [3, Open Problem 4.3], which was first uploaded as a pre-print to the arXiv shortly before Christmas 2017. At the time, the authors of the mentioned work (Daniele D'Angeli, Emanuele Rodaro and the current author) already had some results on the problem, which were, however, not ready for publication and, thus, not included in the mentioned pre-print. However, at the end of January 2018, Dominik Francoeur uploaded a note to the arXiv [Fra18] also giving a proof for a positive answer to the mentioned open problem and approached the current author via email shortly after.

This resulted in a joint paper combining the two approaches and considering further consequences of it, whose first version was uploaded to the arXiv at the beginning of March 2019 and submitted to the *Journal of Algebra*. However, the anonymous reviewer suggested a major revision (and restriction to the main results) of the paper and observed that a step in the proof can be simplified using an argumentation involving the dual automaton. This observation lead to a discussion among the authors and a huge simplification of the proof. Finally, this was the version of the proof that got published in [7] and the proof presented here (mostly) follows the same argument.

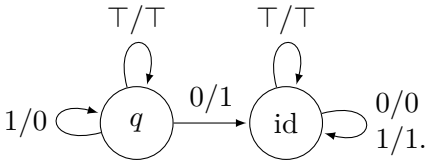
The presented corollaries of the main result and the central dual connection as well as the other results are also joint work with Daniele D'Angeli, Dominik Francoeur and Emanuele Rodaro [7; 8].

1.4.1 Infinite Orbits

In this subsection, we are going to show that every infinite automaton semigroup admits an ω-word with an infinite orbit. Using a standard argument,[33] it is straightforward to see that every infinite automaton semigroup admits a sequence of words with (strictly) growing orbit sizes. However, this sequence might still converge to an ω-word with a finite orbit.

[33] If there is a constant uniformly bounding the orbit size for all words, then there are only finitely many (non-isomorphic) orbital graphs and the semigroup acts faithfully on the finite union of these finite objects. Thus, it must be finite in this case.

Counter Example 1.4.1.1 (compare to [3, Open Problem 4.3])**.** If we add a new letter $\top$ with identity self-loop at all states to the adding machine from Example 0.2.1.4, we obtain the automaton



For this automaton, the word $\top^n 0^n$ (for $n \in \mathbb{N}$) has an orbit of size 2^n. However, the limit[34] $\lim_{n\to\infty} \top^n 0^n$ is $\top^\omega$, which has an orbit of size 1. Of course, there still is 0^ω which has an infinite orbit.[35]

[34] The limit is taken with respect to the standard prefix metric on Σ^∞.

[35] And, of course, we will prove that such a word with an infinite orbit always exists for infinite automaton semigroups.

This example shows that we cannot simply take the limit of a sequence of words with growing orbit sizes to obtain a word with an infinite orbit. In fact, the proof we will use to show our result will be less of a topological nature and rather use the dual automaton.

Generalized Schreier Graphs. To simplify our notation, we fix an arbitrary $\mathscr{S}$-automaton $\mathcal{T} = (Q, \Sigma, \delta)$ for the remainder of this subsection.

Recall that we have $\boldsymbol{p} =_{\mathcal{T}} \boldsymbol{q}$ for $\boldsymbol{p}, \boldsymbol{q} \in Q^*$ if and only if we have $\boldsymbol{p} \circ w = \boldsymbol{q} \circ w$ (or both undefined) for all $w \in \Sigma^\infty$. We will extend this congruence to arbitrary languages. For $L \subseteq \Sigma^\infty$, define the relation $\equiv_L \subseteq Q^* \times Q^*$ by

$$\boldsymbol{p} \equiv_L \boldsymbol{q} \iff \forall w \in L : \boldsymbol{p} \circ w = \boldsymbol{q} \circ w \text{ (or both undefined).}$$

Clearly, $\equiv_{\Sigma^*}$ and $=_{\mathcal{T}}$ are the same congruence and $\equiv_L$ is an equivalence for all $L \subseteq \Sigma^\infty$. In general, however, $\equiv_L$ is not necessarily a congruence (since $\boldsymbol{p} \circ w$ may not be in L although w is). As an equivalence, $\equiv_L$ is coarser than (or equal to) $=_{\mathcal{T}}$ and we write $[\boldsymbol{q}]_L$ for the equivalence class of $\boldsymbol{q} \in Q^*$ with respect to $\equiv_L$.

For a set $K \subseteq Q^*$ of state sequences, we let

$$K/L = \{[\boldsymbol{q}]_L \mid \boldsymbol{q} \in K\}$$

bet the set of equivalence classes (with respect to $\equiv_L$) with a representative in K. On this set, we can define a natural graph structure: its edges are given by

$$\{[p\boldsymbol{q}]_L \xleftarrow{p} [\boldsymbol{q}]_L \mid \boldsymbol{q} \in K, p \in Q \text{ with } p\boldsymbol{q} \in K\}.$$

Just like with orbital graphs, we do not distinguish between K/L as a set and K/L as a graph.

Remark 1.4.1.2. Also like with orbital graphs, we will sometimes draw the edges (and paths) in K/L from right to left instead of from left to right. This is again connected to the nature of the underlying action as a left action. In addition, the label of a finite path

$$[u_\ell]_L \xleftarrow{q_\ell} \cdots \xleftarrow{q_1} [u_0]_L$$

in K/L with $q_1, \ldots, q_\ell \in Q$ is $q_\ell \ldots q_1 \in Q^\ell$ and the label of an infinite path

$$\cdots \xleftarrow{q_2} [u_1]_L \xleftarrow{q_1} [u_0]_L$$

in K/L with $q_1, q_2, \cdots \in Q$ is $\ldots q_2 q_1 \in Q^{-\omega}$.

We say that K/L is *initial* if we have[36] $[\varepsilon]_L \in K/L$. In this case, $[\varepsilon]_L$ is the *initial node* of K/L and an *initial path* is a path starting in this initial node. We say that K/L is *connected* if it is initial and every node can be reached by an initial path. An important case here is that K/L is always (initial and) connected if $K \subseteq Q^*$ is suffix-closed (as a language over Q). Because the out-degree of the nodes in K/L is always bounded by $|Q|$, every infinite connected graph K/L admits an infinite initial and simple path, which shows the following fact.

[36] Obviously, if K contains ε, then K/L is initial.

Fact 1.4.1.3. *Let $K \subseteq Q^*$ be suffix-closed and let $L \subseteq \Sigma^\infty$ be an arbitrary language. If K/L is infinite, it contains an infinite initial and simple path.*

To better understand the structure of K/L, we continue by looking at a few important special cases.

Example 1.4.1.4 (compare to [7, p. 126])**.** We have already seen that $=_{\mathcal{T}}$ and $\equiv_{\Sigma^*}$ are the same relation. Accordingly, Q^+/Σ^* is the same as the automaton semigroup $\mathscr{S}(\mathcal{T})$ and its graph structure is the (left) Cayley graph of $\mathscr{S}(\mathcal{T})$.

For $P \subseteq Q$, P^+/Σ^* is the subsemigroup generated by P in $\mathscr{S}(\mathcal{T})$ and the graph structure is that of the (left) Cayley graph of this subsemigroup.

If $\mathcal{T}$ is a $\mathscr{G}$-automaton, then $\mathrm{Stab}(u) = \mathrm{Stab}^{\mathbb{1}}_{\widetilde{\mathcal{T}}}(u)$ forms the stabilizer H of $u \in \Sigma^\infty$ in $G = \mathscr{G}(\mathcal{T})$. It is straightforward to verify that we have $\boldsymbol{p} \equiv_u \boldsymbol{q}$ if and only if we have $\boldsymbol{p}\,\mathrm{Stab}(u) = \boldsymbol{q}\,\mathrm{Stab}(u)$ in G. This yields that $\widetilde{Q}^*/u$ corresponds to the co-sets G/H and that its graph structure is given by the (left) Schreier graph of G with respect to H.

[37] Recall that $\mathrm{Pre}\,\alpha$ is the set of finite prefixes of an ω-word α.

Finally, if $\mathcal{T}$ is a complete $\mathscr{S}$-automaton and α is an ω-word, then we have[37] $\boldsymbol{p} \equiv_{\mathrm{Pre}\,\alpha} \boldsymbol{q}$ if and only if we have $\boldsymbol{p} \circ \alpha = \boldsymbol{q} \circ \alpha$. Thus, for a set $K \subseteq Q^*$ of state sequences, $K/\mathrm{Pre}\,\alpha$ corresponds to $K \circ \alpha$ and $Q^*/\mathrm{Pre}\,\alpha$ is (isomorphic to) the orbital graph $Q^* \circ \alpha$.

The last example does not work for non-complete automata. We can have that $\boldsymbol{p} \circ \alpha$ and $\boldsymbol{q} \circ \alpha$ are both undefined but that the partial action of $\boldsymbol{p}$ differs from the one of $\boldsymbol{q}$ on some finite prefix of α (and that, thus, we have $\boldsymbol{p} \not\equiv_{\mathrm{Pre}\,\alpha} \boldsymbol{q}$). However, if $K/\mathrm{Pre}\,\alpha$ is connected, we still have that it is finite if and only if $K \circ \alpha$ is finite. We will show this in the proposition after the next lemma.

Lemma 1.4.1.5 (compare to [7, Lemma 2.3])**.** *Let $K \subseteq Q^*$ and, for $L \subseteq \Sigma^*$, define*

$$\vec{L} = \{\alpha \in \Sigma^\omega \mid \textit{infinitely many prefixes of } \alpha \textit{ are in } L\}.$$

If $\boldsymbol{q} \in Q^* \cup Q^{-\omega}$ is the label of a (possibly infinite) initial path in K/L, then we have

$$\mathrm{Suf}\,\boldsymbol{q} \circ \alpha \subseteq K \circ \alpha$$

for all $\alpha \in \vec{L}$.

Proof. Let $q_i \dots q_1$ with $q_1, \dots, q_i \in Q$ be a finite suffix of $\boldsymbol{q}$. Because it is the label of the initial path

$$[q_i \dots q_1]_L \xleftarrow{q_i} \cdots \xleftarrow{q_1} [\varepsilon]_L$$

in K/L, there is some $\boldsymbol{p}_i \in Q^*$ with $\boldsymbol{p}_i \in K$ and $[q_i \dots q_1]_L = [\boldsymbol{p}_i]_L$. It suffices to show $q_i \dots q_1 \circ \alpha = \boldsymbol{p}_i \circ \alpha$ for all $\alpha \in \vec{L}$. Suppose that this is not the case. Then there is already some finite prefix $u \in \Sigma^*$ of α with $q_i \dots q_1 \circ u \neq \boldsymbol{p}_i \circ u$ (including the case that one is defined while the other one is not). Since α is from $\vec{L}$, there always is a prefix u' of α longer than u with $u' \in L$. Since we have $q_i \dots q_1 \equiv_L \boldsymbol{p_i}$, we must have $q_i \dots q_1 \circ u' = \boldsymbol{p}_i \circ u'$ (or both undefined), which is a contradiction. □

An important special case of Lemma 1.4.1.5 is to choose $L = \mathrm{Pre}\,\alpha$ as the set of finite prefixes of some ω-word α. In this case, we have $\vec{L} = \overrightarrow{\mathrm{Pre}\,\alpha} = \{\alpha\}$.

Proposition 1.4.1.6 (compare to [7, Proposition 2.4]). *Let $K \subseteq Q^*$ and $\alpha \in \Sigma^\omega$ such that $K/\operatorname{Pre}\alpha$ is connected. Then, we have*

$$|K/\operatorname{Pre}\alpha| = \infty \iff |K \circ \alpha| = \infty.$$

Proof. If $K \circ \alpha$ is infinite, there are infinitely many $\boldsymbol{q}_0, \boldsymbol{q}_1, \dots \in K$ such that all $\boldsymbol{q}_i \circ \alpha$ are (defined but) pairwise different, i. e. for $i \neq j$, we have $\boldsymbol{q}_i \circ \alpha \neq \boldsymbol{q}_j \circ \alpha$ (but both defined). Thus, there is some finite prefix u of α with $\boldsymbol{q}_i \circ u \neq \boldsymbol{q}_j \circ u$, which shows $\boldsymbol{q}_i \not\equiv_{\operatorname{Pre}\alpha} \boldsymbol{q}_j$ for all $i \neq j$ and, therefore, that $K/\operatorname{Pre}\alpha$ is infinite.

For the other direction, assume that $K/\operatorname{Pre}\alpha$ is infinite. Since it is also connected, it contains some infinite simple and initial path

$$\cdots \xleftarrow{p_3} [p_2 p_1]_{\operatorname{Pre}\alpha} \xleftarrow{p_2} [p_1]_{\operatorname{Pre}\alpha} \xleftarrow{p_1} [\varepsilon]_{\operatorname{Pre}\alpha}\ .$$

Since the path is simple, we have that all $[p_i \dots p_1]_{\operatorname{Pre}\alpha}$ are pairwise distinct.

We first show that all $p_i \dots p_1 \circ \alpha$ are defined using a contradiction. Suppose that there is some i_0 such that $p_{i_0} \dots p_1 \circ \alpha$ is undefined. There must be some prefix ua of α with $a \in \Sigma$ such that $p_{i_0} \dots p_1 \circ ua$ is undefined. Then, $p_j \dots p_1 \circ ua$ must also be undefined for all $j \geq i_0$. This means that all infinitely many $p_i \dots p_1 \circ u$ with $i \geq i_0$ must be pairwise different (as we have $p_i \dots p_1 \not\equiv_{\operatorname{Pre}\alpha} p_j \dots p_1$ for $i \neq j$), which is not possible.

We obtain that all $p_i \dots p_1 \circ \alpha$ are defined and, by Lemma 1.4.1.5 (with $L = \operatorname{Pre}\alpha$), they must be in $K \circ \alpha$. Additionally, they must also be pairwise different since $p_i \dots p_1$ already acts differently to $p_j \dots p_1$ on some finite prefix of α (for $i \neq j$). This shows that $K \circ \alpha$ is infinite. □

An important (and elegant!) special case of the last proposition is when K is given as the (suffix-closed) set of finite suffixes of some left-infinite sequence over Q.

Corollary 1.4.1.7 ([7, Corollary 2.5]). *For $\pi \in Q^{-\omega}$ and $\alpha \in \Sigma^\omega$, we have*

$$|\operatorname{Suf}\pi/\operatorname{Pre}\alpha| = \infty \iff |\operatorname{Suf}\pi \circ \alpha| = \infty$$

The proof for our main result is of a dual nature at its core. It is straight-forward that we can extend the relation $\equiv_L$ and the graph K/L to the dual $\partial\mathcal{T}$ of $\mathcal{T}$. For a language $K \subseteq Q^\infty$, we define the relation $\equiv_K \subseteq \Sigma^* \times \Sigma^*$ by

$$u \equiv_K v \iff \forall \boldsymbol{p} \in K : u \circ_\partial \boldsymbol{p} = v \circ_\partial \boldsymbol{p} \text{ (or both undefined).}$$

That this relation is to be understood with respect to the dual can be seen from the fact that K is a subset of Q^∞ and not of Σ^∞ (as previously in the definition of $\equiv_L$). Now, for $L \subseteq \Sigma^*$, we can define L/K as those classes with respect to $\equiv_K$ that have a representative in L and add edges to obtain the graph structure: for all $u \in L$ and all $a \in \Sigma$ with $au \in L$, we add an a-labeled edge from the class of u to the class of au.

However, instead of dealing with K/L with respect to $\mathcal{T}$ and L/K with respect to $\partial\mathcal{T}$, we can directly define a dual object to K/L only working with $\mathcal{T}$.

Dual Schreier Graphs. For all $K \subseteq Q^* \cup Q^{-\omega}$, we define the relation $\sim_K \subseteq \Sigma^* \times \Sigma^*$ by

$$u \sim_K v \iff \forall \boldsymbol{q} \in K : \boldsymbol{q} \cdot u = \boldsymbol{q} \cdot v \text{ (or both undefined).}$$

Just like $\equiv_L$, this is clearly an equivalence and we write $[u]_K$ for the class of u with respect to $\sim_K$. For a language $L \subseteq \Sigma^*$, we define

$$K\backslash L = \{[u]_K \mid u \in L\}$$

as the set of classes with a representative in L. Again, we can define a natural graph structure on this set whose edges are given by

$$\{[u]_K \xrightarrow{a} [ua]_K \mid u \in L, a \in \Sigma \text{ with } ua \in L\}$$

and we will not distinguish between the set and the graph notationally.

It is not difficult to see[38] that we have $u \sim_K v$ if and only if we have $u \equiv_{\partial K} v$ for all $u, v \in \Sigma^*$. From this, we obtain that $K\backslash L$ and $\partial L/\partial K$ are basically the same object:

[38] We only have to exploit the connection $\boldsymbol{p} \cdot u = \partial(\partial u \circ_\partial \partial \boldsymbol{p})$.

Fact 1.4.1.8. *The map*

$$K\backslash L \to \partial L/\partial K$$
$$[u]_K \mapsto [\partial u]_{\partial K}$$

(where $\partial L/\partial K$ is to be understood with respect to the dual automaton $\partial\mathcal{T}$) is a well-defined bijection and graph isomorphism.

Of course, we can also transfer the notations of being initial or connected to $K\backslash L$: it is *initial* if we have $[\varepsilon]_K \in K\backslash L$, in which case we call $[\varepsilon]_K$ the *initial node* of $K\backslash L$ and any path starting there an *initial path*. The graph is *connected* if it is initial and every node is reachable from $[\varepsilon]_K$. Again, we have the important special case that $K\backslash L$ is connected if $L \subseteq \Sigma^*$ is prefix-closed. Since the out-degree of a node in $K\backslash L$ is always bounded by $|\Sigma|$, we obtain an analogue of Fact 1.4.1.3.

Fact 1.4.1.9. *Let $K \subseteq Q^* \cup Q^{-\omega}$ be arbitrary and let $L \subseteq \Sigma^*$ be prefix-closed. If $K\backslash L$ is infinite, it contains an infinite initial and simple path.*

Finally, we can clearly also define $\sim_L$ and $L\backslash K$ with respect to $\partial\mathcal{T}$. For $L \subseteq \Sigma^* \cup \Sigma^{-\omega}$, we let $\sim_L \subseteq Q^* \times Q^*$ be given by

$$\boldsymbol{p} \sim_L \boldsymbol{q} \iff \forall u \in L : u \cdot_\partial \boldsymbol{p} = u \cdot_\partial \boldsymbol{q} \text{ (or both undefined)}$$

and define $L\backslash K$ for $K \subseteq Q^*$ as those classes with respect to $\sim_L$ that have a representative in K. To define the graph structure of $L\backslash K$, we add a q-labeled edge from the class of $\boldsymbol{p}$ to the class of $\boldsymbol{p}q$ for all $\boldsymbol{p} \in K$ and $q \in Q$ with $\boldsymbol{p}q \in K$.

Remark 1.4.1.10. While the many objects K/L, $K\backslash L$, L/K and $L\backslash K$ might seem confusing, the reader can rest assured that we will actually mostly be working with the ones defined with respect to $\mathcal{T}$; only once (and only briefly), we will also need those defined with respect to $\partial\mathcal{T}$ in our main proof.

The Main Proof. We have now all objects at hand that we need in order to prove our main result. In fact, we will first prove a result on the dual nature between K/L and $K\backslash L$. The underlying idea here it to generalize the fact that an $\mathscr{S}$-automaton generates an infinite semigroup if and only if its dual does.[39]

Proposition 1.4.1.11 (see [7, Proposition 3.1])**.** *Let $K \subseteq Q^*$ be suffix-closed and let $L \subseteq \Sigma^*$ be prefix-closed. Then, we have:*

$$|K/L| = \infty \iff \infty = |K\backslash L|$$

Proof. We only show one direction as the other one then follows by duality:

$$\infty = |K\backslash L| = |\partial L/\partial K| \implies |\partial L\backslash\partial K| = |K/L| = \infty$$

(where we have used $\infty = |K/L| \implies |K\backslash L| = \infty$ with respect to the dual automaton).

We show this direction by using contraposition. So, assume that we have $|K\backslash L| = C < \infty$. Then, for all $\boldsymbol{q} \in K$, we have $|\boldsymbol{q} \cdot L| \leq C$ and, thus, that the size of the $\mathscr{S}$-automaton with state set $\boldsymbol{q} \cdot L$, alphabet Σ and the transitions

$$\{\boldsymbol{q} \cdot u \xrightarrow{a/\boldsymbol{q} \cdot u \circ a} \boldsymbol{q} \cdot ua \mid ua \in L, \boldsymbol{q} \cdot ua \text{ defined}\}$$

is at most C. Clearly, this automaton fully describes the partial action of $\boldsymbol{q} \in K$ on words from L.[40] However, there are only finitely many such automata (that are non-isomorphic) and, therefore, we can only have finitely many different partial actions of elements from K on words from L. In other words, K/L must be finite. □

Corollary 1.4.1.12 ([SV11, Proposition 2.2])**.** *We have:*

$$|\mathscr{S}(\mathcal{T})| = \infty \iff \infty = |\mathscr{S}(\partial\mathcal{T})|$$

Proof. We have

$$|\mathscr{S}(\mathcal{T})| = |Q^*/\Sigma^*| = \infty \iff \infty = |Q^*\backslash\Sigma^*| = |\Sigma^*/Q^*| = |\mathscr{S}(\partial\mathcal{T})|$$

where we have used that $\equiv_{\Sigma^*}$ is the equality in $\mathscr{S}(\mathcal{T})$ (for the first equality), Proposition 1.4.1.11 (for the equivalence), Fact 1.4.1.8 (for the second equality) and, finally, that $\equiv_{Q^*}$ is the equality in $\mathscr{S}(\partial\mathcal{T})$ (for the last equality). □

While also interesting on its own, Proposition 1.4.1.11 is the central part for proving that there is a single ω-word with an infinite K-orbit if K is suffix-closed and forms an infinite subset in an automaton semigroup.

Theorem 1.4.1.13 ([7, Theorem 3.2])**.** *Let $K \subseteq Q^*$ be suffix-closed for the $\mathscr{S}$-automaton $\mathcal{T} = (Q, \Sigma, \delta)$. Then, we have:*

$$K \text{ is infinite in } \mathscr{M}(\mathcal{T}) \iff \exists \alpha \in \Sigma^\omega : |K \circ \alpha| = \infty$$

[39] This result seems to be first given in [SV11, Proposition 2.2] (see also [Akh+12, Proposition 10]), where it is stated that the result was also independently obtained by Šunić. Furthermore, it is already present implicitly in the proof of [Nek05, Lemma 1.10.6].

[40] Since L is prefix-closed, the stated automaton contains all runs of $\mathcal{T}^{|\boldsymbol{q}|}$ starting in $\boldsymbol{q}$ whose input is from L.

Proof. If the image of K in $\mathscr{M}(\mathcal{T})$ is of finite size C, then the size of all K-orbits $K \circ \alpha$ with $\alpha \in \Sigma^\omega$ is clearly also bounded by C.

If, on the other hand, the image of K in $\mathscr{M}(\mathcal{T})$ is infinite, we have that K/Σ^* is infinite since $\equiv_{\Sigma^*}$ is the equality in $\mathscr{M}(\mathcal{T})$ and it is connected because K is suffix-closed. By Fact 1.4.1.3, there is an infinite initial and simple path in K/Σ^* and we let $\pi \in Q^{-\omega}$ denote its label. From Lemma 1.4.1.5, we obtain $\operatorname{Suf} \pi \circ \alpha \subseteq K \circ \alpha$ for all $\alpha \in \Sigma^\omega$. Thus, it suffices to show $|\operatorname{Suf} \pi \circ \alpha| = \infty$ for some $\alpha \in \Sigma^\omega$.

Clearly, the infinite initial and simple path belonging to π also exists in the subgraph $\operatorname{Suf} \pi/\Sigma^*$ (and it remains initial and simple). Thus, we have that $\operatorname{Suf} \pi/\Sigma^*$ is infinite. Now, Proposition 1.4.1.11 states that $\operatorname{Suf} \pi \backslash \Sigma^*$ must also be infinite, which implies that it must contain an infinite initial and simple path (by Fact 1.4.1.9) whose label we denote by $\alpha \in \Sigma^\omega$. Again, this path must also exist in the subgraph $\operatorname{Suf} \pi \backslash \operatorname{Pre} \alpha$, which, thus, must be infinite. Applying Proposition 1.4.1.11 a second time yields that $\operatorname{Suf} \pi / \operatorname{Pre} \alpha$ must be infinite and, by Corollary 1.4.1.7, that $\operatorname{Suf} \pi \circ \alpha \subseteq K \circ \alpha$ is infinite as well. □

While making a statement about automaton monoids instead of automaton semigroups allows for a more elegant formulation in Theorem 1.4.1.13, we clearly have that $K \setminus \{\varepsilon\}$ is infinite in $\mathscr{S}(\mathcal{T})$ is and only if K is infinite in $\mathscr{M}(\mathcal{T})$ for $K \subseteq Q^*$.

Corollaries. The formulation of Theorem 1.4.1.13 is very general. Obviously, its most interesting special case is to take K as Q^*, which (together with Fact 0.2.3.1) yields than an automaton semigroup is infinite if and only if it admits an ω-word with an infinite orbit.

Corollary 1.4.1.14 ([7, Corollary 3.3])**.** *We have*

$$|\mathscr{S}(\mathcal{T})| = \infty \iff \exists \alpha \in \Sigma^\omega : |Q^* \circ \alpha| = \infty$$

for the $\mathscr{S}$-automaton $\mathcal{T} = (Q, \Sigma, \delta)$.

This connection allows for a re-formulation of the finiteness problem for automaton (semigroups and) groups, which we will discuss in Subsection 2.3.3 and use in Subsection 2.4.3. Together with Fact 0.3.2.2, it also yields the following well-known connection between the finiteness of the semigroup and the group generated by a $\mathscr{G}$-automaton.

Fact 1.4.1.15. *If $\mathcal{T} = (Q, \Sigma, \delta)$ is a $\mathscr{G}$-automaton, we have:*

$$|\mathscr{S}(\mathcal{T})| = \infty \iff |\mathscr{G}(\mathcal{T})| = \infty$$

Proof. One direction immediately follows from the functional view of automaton semigroups, which interprets $\mathscr{S}(\mathcal{T})$ as a subset of $\mathscr{G}(\mathcal{T})$. If, on the other hand, $\mathscr{G}(\mathcal{T})$ is infinite, there is some ω-word $\alpha \in \Sigma^\omega$ with an infinite orbit $\widetilde{Q}^* \circ \alpha$ by Corollary 1.4.1.14 (applied to $\widetilde{\mathcal{T}}$). From Fact 0.3.2.2, we obtain that $Q^* \circ \alpha$ and, thus, $\mathscr{S}(\mathcal{T})$ must also be infinite. □

However, Theorem 1.4.1.13 also covers (finitely generated) subsemigroups of automaton semigroups. Here, it is worthwhile to recall that the class of automaton semigroups is not closed under taking subsemigroups.[41]

[41] ...as we have seen in Section 1.2 that the monogenic free monoid is an automaton semigroup while the monogenic free semigroup – one of its subsemigroups – is not.

Corollary 1.4.1.16 ([7, Corollary 3.7])**.** *Let $\boldsymbol{P} \subseteq Q^+$ be a finite set of state sequences for the $\mathscr{S}$-automaton $\mathcal{T} = (Q, \Sigma, \delta)$. Then, the subsemigroup given by $\boldsymbol{P}^+$ in $\mathscr{S}(\mathcal{T})$ is infinite if and only if there is some $\alpha \in \Sigma^\omega$ with $|\boldsymbol{P}^* \circ \alpha| = \infty$.*

Proof. We may assume $\boldsymbol{P} = P \subseteq Q$, i. e. that P only contains single states, since we can replace $\mathcal{T}$ by the union of $\mathcal{T}$ and suitable powers of $\mathcal{T}$. With this assumption, P^* is a suffix-closed language over Q. The subsemigroup given by P^+ in $\mathscr{S}(\mathcal{T})$ is infinite if and only if P^* is infinite in $\mathscr{M}(\mathcal{T})$ and, by Theorem 1.4.1.13, this is the case if and only if there is some $\alpha \in \Sigma^\omega$ with $|P^* \circ \alpha| = \infty$. □

We even obtain results for some non-finitely generated subsemigroups from Theorem 1.4.1.13. As an example, we will apply it to principal left ideals.

Semigroup Ideals. The *principal left ideal*[42] of a semigroup element $s \in S$ is $S^1 s = \{s's \mid s' \in S^1\}$. Accordingly, the *principal right ideal* of s is $sS^1 = \{ss' \mid s' \in S^1\}$ and the *principal (two-sided) ideal* of s is $S^1 s S^1 = \{s'ss'' \mid s', s'' \in S^1\}$.

By definition, left, right and two-sided ideals are always subsemigroups. However, in general, they are not finitely generated. For example, the principal left ideal $\{a, b\}^* a$ in the free semigroup $\{a, b\}^+$ (which is an automaton semigroup by [Cai09, Proposition 4.1]) is not finitely generated. Still, using Theorem 1.4.1.13, we can show an analogue to Corollary 1.4.1.16 for the case of principal left ideals.

[42] For the definition of a (semigroup) ideal, see page 30.

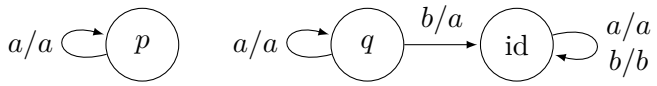
Corollary 1.4.1.17 ([7, Corollary 3.9])**.** *Let $\boldsymbol{p} \in Q^+$ for the $\mathscr{S}$-automaton $\mathcal{T} = (Q, \Sigma, \delta)$. For the left principal ideal of $\boldsymbol{p}$ in $\mathscr{S}(\mathcal{T})$, we have:*

$$Q^*\boldsymbol{p} = \{\boldsymbol{qp} \mid \boldsymbol{q} \in Q^*\} \text{ is in } \mathscr{S}(\mathcal{T}) \text{ infinite} \iff \exists \alpha \in \Sigma^\omega : |Q^*\boldsymbol{p} \circ \alpha| = \infty$$

Proof. By replacing $\mathcal{T}$ with $\mathcal{T} \uplus \mathcal{T}^{|\boldsymbol{p}|}$, we may assume $\boldsymbol{p} = p \in Q$. This turns $Q^*p \cup \{\varepsilon\}$ into a suffix-closed language over Q. Now, the left principal ideal Q^*p in $\mathscr{S}(\mathcal{T})$ is infinite if and only if $Q^*p \cup \{\varepsilon\}$ is infinite in $\mathscr{M}(\mathcal{T})$, which, by Theorem 1.4.1.13, holds if and only if there is some ω-word $\alpha \in \Sigma^\omega$ with $|(Q^*p \cup \{\varepsilon\}) \circ \alpha| = \infty$. Obviously, this is the case if and only if $Q^*p \circ \alpha$ is infinite. □

We cannot extend our result to all (non-finitely generated) subsemigroups, however, as the following counter-example shows. In fact, we can have infinite principal right or two-sided ideals which do not admit a single ω-word with an infinite orbit.

Counter Example 1.4.1.18 (see [7, Counter-Example 3.10])**.** If we choose $\mathcal{T} = (Q, \Sigma, \delta)$ as the $\mathscr{S}$-automaton



and consider $K = pQ^*$, then we clearly have either $K \circ \alpha = \emptyset$ or $K \circ \alpha = \{a^\omega\}$ for all $\alpha \in \Sigma^\omega$. However, K is infinite in $\mathscr{S}(\mathcal{T})$ as we have $pq^i \neq pq^j$ in $\mathscr{S}(\mathcal{T})$ for all $i < j$. This is the case because we have that $pq^i \circ b^j = p \circ a^i b^{j-i}$ is undefined while $pq^j \circ b^j$ is a^j (and, thus, defined). This also shows that Q^*pQ^* is infinite in $\mathscr{S}(\mathcal{T})$. Since a^ω is fixed by the partial actions of all states, we also have either $Q^*pQ^* \circ \alpha = \emptyset$ or $Q^*pQ^* \circ \alpha = \{\alpha^\omega\}$ for all $\alpha \in \Sigma^\omega$.

A Corollary of Proposition 1.4.1.11. We conclude this subsection with another corollary of Proposition 1.4.1.11 about the orbit and the dual orbit belonging to a single left infinite sequence and a single ω-word.

Corollary 1.4.1.19 (see [7, Corollary 3.11])**.** *For $\pi \in Q^{-\omega}$ and $\alpha \in \Sigma^\omega$, we have*

$$|\operatorname{Suf}\pi \circ \alpha| = \infty \iff \infty = |\pi \cdot \operatorname{Pre}\alpha| = |\operatorname{Suf}\partial\alpha \circ_\partial \partial\pi|$$

Proof. The equality on the right hand side is due to the bijection from Fact 0.3.2.4 and we have:

$$\begin{aligned}
&|\operatorname{Suf}\pi \circ \alpha| = \infty \\
\iff &|\operatorname{Suf}\pi / \operatorname{Pre}\alpha| = \infty && \text{(by Corollary 1.4.1.7)} \\
\iff &|\operatorname{Suf}\pi \backslash \operatorname{Pre}\alpha| = \infty && \text{(by Proposition 1.4.1.11)} \\
\iff &|\partial\operatorname{Pre}\alpha / \partial\operatorname{Suf}\pi| = \infty && \text{(by Fact 1.4.1.8)} \\
\iff &|\operatorname{Suf}\partial\alpha / \operatorname{Pre}\partial\pi| = \infty && \text{(since } \partial\operatorname{Pre}\alpha = \operatorname{Suf}\partial\alpha \text{ and } \partial\operatorname{Suf}\pi = \operatorname{Pre}\partial\pi\text{)} \\
\iff &|\operatorname{Suf}\partial\alpha \circ_\partial \partial\pi| = \infty && \text{(by Corollary 1.4.1.7, applied to } \partial\mathcal{T}\text{).}
\end{aligned}$$

□

1.4.2 Orbits of Periodic and Non-Periodic Words

Torsion and the Dual Orbit. We can derive further interesting corollaries directly from Proposition 1.4.1.11. For example, we can give a simple proof to show that an element of an automaton semigroup has torsion if and only if the orbit of the corresponding periodic word has a finite orbit under the action of the dual automaton. This connection had previously been shown for groups [DR16, Theorem 3] and, independently, for reversible $\mathscr{G}$-automata [KPS18, Proposition 7].

First, we link the finiteness of the image of $\operatorname{Suf} q^{-\omega}$ in the automaton monoid/semigroup to the property that $\boldsymbol{q}$ has torsion.

Fact 1.4.2.1. *For an $\mathscr{S}$-automaton $\mathcal{T} = (Q, \Sigma, \delta)$ and $\boldsymbol{q} \in Q^*$, we have:*

$$\operatorname{Suf}\boldsymbol{q}^{-\omega} \text{ is infinite in } \mathscr{M}(\mathcal{T}) \iff \boldsymbol{q} \text{ has torsion in } \mathscr{S}(\mathcal{T})$$

Proof. If $\boldsymbol{q}$ has no torsion in $\mathscr{S}(\mathcal{T})$, we have that $\operatorname{Suf}\boldsymbol{q}^{-\omega}$ is infinite in $\mathscr{M}(\mathcal{T})$ since we must have that already all $\boldsymbol{q}^i \in \operatorname{Suf}\boldsymbol{q}^{-\omega}$ with $i \geq 0$ must be pairwise distinct in $\mathscr{M}(\mathcal{T})$.

For the other direction, suppose that we have $\boldsymbol{q}^i = \boldsymbol{q}^j$ in $\mathscr{S}(\mathcal{T})$ for some $0 < i < j$ and consider an arbitrary element $\boldsymbol{r}$ of $\operatorname{Suf}\boldsymbol{q}^{-\omega}$. It must be of the form $\boldsymbol{r} = \boldsymbol{p}\boldsymbol{q}^k$ where $k \in \mathbb{N}$ and $\boldsymbol{p} \in Q^*$ is a suffix of $\boldsymbol{q}$ (and, thus, from a finite set). For $k \geq j$, we have $\boldsymbol{p}\boldsymbol{q}^k =_{\mathcal{T}} \boldsymbol{p}\boldsymbol{q}^{k-(j-i)}$. Thus, only values of k with $k < j$ can yield elements in $\operatorname{Suf}\boldsymbol{q}^{-\omega}$ that are distinct in $\mathscr{M}(\mathcal{T})$. □

Now, proving the actual connection between a torsion element in the dual and the finiteness of the corresponding orbit boils down to applying Proposition 1.4.1.11.

Theorem 1.4.2.2 (dual version of [7, Theorem 3.12])**.** *Let $u \in \Sigma^+$ for an $\mathscr{S}$-automaton $\mathcal{T} = (Q, \Sigma, \delta)$. Then, the statements*

1. *∂u has torsion in $\mathscr{S}(\partial\mathcal{T})$.*
2. *The orbit $Q^* \circ u^\omega$ is finite.*
3. *The orbit $Q^* \circ vu^\omega$ is finite for all $v \in \Sigma^*$.*

are equivalent.

Proof. For the second equivalence, observe that we have $Q^* \circ vu^\omega \subseteq \Sigma^{|u|}(Q^* \circ u^\omega)$ for all $v \in \Sigma^*$. Thus, if $Q^* \circ u^\omega$ is finite, so are all $Q^* \circ vu^\omega$ with $v \in \Sigma^*$ (and the converse is trivial).

Regarding the first equivalence, we have

$$
\begin{aligned}
& |Q^* \circ u^\omega| < \infty \\
\iff & |Q^* / \operatorname{Pre} u^\omega| < \infty && \text{(by Proposition 1.4.1.6)} \\
\iff & |Q^* \setminus \operatorname{Pre} u^\omega| < \infty && \text{(by Proposition 1.4.1.11)} \\
\iff & |\partial \operatorname{Pre} u^\omega / Q^*| < \infty && \text{(by Fact 1.4.1.8)} \\
\iff & |\operatorname{Suf}(\partial u)^{-\omega} / Q^*| < \infty && \text{(since } \partial \operatorname{Pre} u^\omega = \operatorname{Suf}(\partial u)^{-\omega}\text{)} \\
\iff & \operatorname{Suf}(\partial u)^{-\omega} \text{ is finite in } \mathscr{M}(\partial\mathcal{T}) && (\equiv_{Q^*} \text{ is the equality in } \mathscr{S}(\partial\mathcal{T})) \\
\iff & \partial u \text{ has torsion in } \mathscr{S}(\partial\mathcal{T}) && \text{(by Fact 1.4.2.1, applied to } \partial\mathcal{T}\text{).}
\end{aligned}
$$

□

Later on, it will be useful to also have a dual formulation of Theorem 1.4.2.2 at hand. This formulation directly follows from swapping the roles of $\mathcal{T}$ and $\partial\mathcal{T}$.

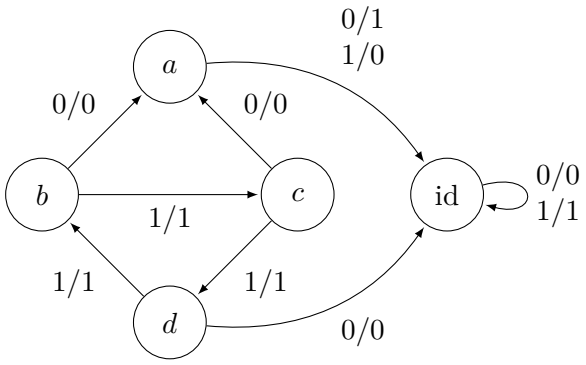
Theorem 1.4.2.3 ([7, Theorem 3.12], dual version of Theorem 1.4.2.2)**.** *Let $\boldsymbol{q} \in Q^+$ for an $\mathscr{S}$-automaton $\mathcal{T} = (Q, \Sigma, \delta)$. Then, the statements*

1. *$\partial\boldsymbol{q}$ has torsion in $\mathscr{S}(\mathcal{T})$.*
2. *The orbit $\Sigma^* \circ_\partial \boldsymbol{q}^\omega$ of $\boldsymbol{q}^\omega$ under the action of the dual of $\mathcal{T}$ is finite.*
3. *The orbit $\Sigma^* \circ_\partial \boldsymbol{pq}^\omega$ of $\boldsymbol{pq}^\omega$ under the action of the dual of $\mathcal{T}$ is finite for all $\boldsymbol{p} \in Q^*$.*

are equivalent.

Non-Periodicity. We have seen (in Corollary 1.4.1.14) that every infinite automaton semigroup admits an ω-word with an infinite orbit. However, the proof is purely existential and does not allow us to extract much information about the nature of the word. In fact, we can prove that it does not have certain structural properties. Namely, we will exploit the connection from Theorem 1.4.2.3 in the next counter example to see that, in general, it cannot be periodic or ultimately periodic.

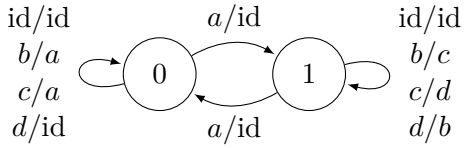
Counter Example 1.4.2.4 (compare to [8, Counter-Example 5.7])**.** Let $\mathcal{T}$ be a $\mathscr{G}$-automaton generating an infinite torsion group $\mathscr{G}(\mathcal{T})$. For example, we can choose $\mathcal{T}$ as the automaton



generating the Grigorchuk group, which is an infinite torsion group (see, e. g. [Nek05, Theorem 1.6.1]).

Since $\mathscr{G}(\mathcal{T})$ is infinite, we also have that $\mathscr{S}(\mathcal{T})$ and, thus, $\mathscr{S}(\partial\mathcal{T})$ are infinite (by Fact 1.4.1.15 and Corollary 1.4.1.12). Now assume that we have an ultimately periodic word $\boldsymbol{pq}^\omega$ with $\boldsymbol{p} \in Q^*$ and $\boldsymbol{q} \in Q^+$ with an infinite orbit $\Sigma^* \circ_\partial \boldsymbol{pq}$ (under the action of the dual). Since we have $\Sigma^* \circ_\partial \boldsymbol{pq}^\omega \subseteq Q^{|\boldsymbol{p}|}(\Sigma^* \circ_\partial \boldsymbol{q}^\omega)$, we obtain that already the orbit $\Sigma^* \circ_\partial \boldsymbol{q}^\omega$ of the periodic word $\boldsymbol{q}^\omega$ must be infinite. Now, Theorem 1.4.2.3 yields that $\partial\boldsymbol{q}$ must have torsion in $\mathscr{S}(\mathcal{T})$ and, thus, in $\mathscr{G}(\mathcal{T})$, which constitutes a contradiction.

This means that, for example, the complete and reversible dual $\partial\mathcal{T}$



of the above automaton generates an infinite semigroup in which all periodic and ultimately periodic words have a finite orbit.

Finite Orbits. So far, we have mostly looked at infinite orbits. However, Proposition 1.4.1.11 can also be used to derive results about finite orbits. In contrast to Counter-Example 1.4.2.4, we can show in this setting that every automaton semigroup that admits an ω-word with a finite orbit already admits an ultimately periodic ω-word with a finite orbit. In the case of a complete and reversible automaton, it even admits a periodic ω-word with an infinite orbit.

Proposition 1.4.2.5 ([8, Proposition 4.1])**.** *For any $\mathscr{S}$-automaton $\mathcal{T} = (Q, \Sigma, \delta)$, we have*

$$\exists \alpha \in \Sigma^\omega : |Q^* \circ \alpha| < \infty \implies \exists u \in \Sigma^*, v \in \Sigma^+ : |Q^* \circ uv^\omega| < \infty.$$

In fact, u and v can be chosen in such a way that they both contain all letters that appear infinitely often in α.

If $\mathcal{T}$ is a complete and reversible $\mathscr{S}$-automaton, we even have

$$\exists \alpha \in \Sigma^\omega : |Q^* \circ \alpha| < \infty \implies \exists v \in \Sigma^+ : |Q^* \circ v^\omega| < \infty.$$

Proof (adapted from the proof of [8, Proposition 4.1]). Suppose we have $|Q^* \circ \alpha| < \infty$ for the ω-word $\alpha = a_1 a_2 \dots$ with $a_1, a_2, \dots \in \Sigma$ and the $\mathscr{S}$-automaton $\mathcal{T}$. By Proposition 1.4.1.6, this is equivalent to $|Q^* / \operatorname{Pre} \alpha| < \infty$, which, in turn, is equivalent to $|Q^* \backslash \operatorname{Pre} \alpha| < \infty$ by Proposition 1.4.1.11. Thus, if we read the initial path labeled by α in the graph $Q^* \backslash \operatorname{Pre} \alpha$, there is some node U that is visited infinitely often. Let $I = \{i \in \mathbb{N} \mid [a_1 \dots a_i]_{Q^*} = U\}$ be the corresponding infinite index set. Choose some arbitrary elements $i, j \in I$ with $i < j$ and let $u = a_1 \dots a_i$ and $v = a_{i+1} \dots a_j$. In fact, we can choose i and j large enough so that u and v both contain all letters that appear infinitely often in α at least once. From their definition, u labels an initial path from $[\varepsilon]_{Q^*}$ to $[u]_{Q^*} = U$ in $Q^* \backslash \operatorname{Pre} \alpha$ and v labels a loop at $[u]_{Q^*} = [uv]_{Q^*} = [uv^2]_{Q^*} = \dots = U$. This initial path and the loop form the graph $Q^* \backslash \operatorname{Pre} uv^\omega$, which is, therefore, finite. Again, by Proposition 1.4.1.11, we obtain $|Q^* / \operatorname{Pre} uv^\omega| < \infty$, which is equivalent to $|Q^* \circ uv^\omega| < \infty$ by Proposition 1.4.1.6.

If $\mathcal{T}$ is additionally complete and reversible, there is a surjective function $Q^* \circ uv^\omega \to Q^* \circ v^\omega$, which shows $|Q^* \circ v^\omega| \leq |Q^* \circ uv^\omega| < \infty$. This function maps a word $w\beta \in Q^* \circ uv^\omega$ with prefix w of length $|w| = |u|$ to β. Clearly, $w\beta \in Q^* \circ uv^\omega$ implies $\beta \in Q^* \circ v^\omega$ and, to show that the function is surjective, consider an arbitrary element $\beta \in Q^* \circ v^\omega$. Then, there is some $\boldsymbol{q} \in Q^*$ with $\boldsymbol{q} \circ v^\omega = \beta$. Since $\mathcal{T}$ is complete and reversible, there is some $\boldsymbol{p} \in Q^{|\boldsymbol{q}|}$ with $\boldsymbol{p} \cdot u = \boldsymbol{q}$ (as the map $\cdot u$ is a length-preserving permutation in this case by Fact 0.3.1.2 and we can choose $\boldsymbol{p}$ as the pre-image of $\boldsymbol{q}$). This yields the cross diagram

$$\begin{array}{ccccc} & u & & v^\omega & \\ \boldsymbol{p} & \downarrow & \boldsymbol{q} & \downarrow & \\ & w & & \beta & \end{array}$$

for $w = \boldsymbol{p} \circ u$ and, thus, that $w\beta \in Q^* \circ uv^\omega$ is a pre-image of β for our function. □

2 Decision Problems

2.1 Word Problem

While, so far, we have mostly studied structural properties of automaton structures, we move our attention to the study of algorithmic problems in this chapter. Among other reasons, automata are interesting as inputs to algorithms because they sometimes allow for a finite description of algebraic structures which cannot be finitely described using the classical presentation with generators and relations.[43] Together with many undecidability results, this indicates that automaton structures show a complex behavior also algorithmically.

In this light, it might be surprising that the first – and probably most fundamental – of Dehn's three fundamental problems [Deh11][44] in algorithmic group theory, the word problem, is decidable for automaton structures. The word problem of a finitely generated group asks whether a given word in the generators represents the neutral element of the group. This problem naturally generalizes to semigroups (and monoids) where the input consists of two words over the generators and the question is whether both represent the same semigroup element. Additionally, there are also so-called uniform versions of the word problem where the group or semigroup itself is an additional part of the input in a suitable (usually finite) representation.

In the setting of automaton structures, the words over the generators are state sequences and the representation for the uniform versions is obviously the generating automaton. While there are other ways to solve the word problem for automaton structures,[45] it was Steinberg who first introduced a nondeterministic "guess and check" algorithm requiring only linear space to solve the word problem of automaton groups [Ste15, section 3]. It turns out that this (rather straight-forward) algorithm is actually an algorithm for automaton semigroups (even partial ones) and we will briefly discuss and analyze this in the first subsection. As a byproduct of the algorithm, we can re-prove a result by Cain giving an upper bound on the length of a shortest word on which two state sequences representing different elements of an automaton semigroup must act differently [Cai09, Corollary 3.5]. Cain also asked the question whether this bound can be improved [Cai09, Open problem 3.6] and we give an explicit lower bound construction for the length and discuss why a significant improvement is impossible (under common assumptions from complexity theory).

In addition to stating the "guess and check" algorithm, Steinberg also conjectured that there is an automaton group whose word problem is PSpace-complete [Ste15, Question 5] (see also [AIM07, section 2, 6.]). Proving this conjecture is the main result of this section (which we will present in Subsection 2.1.2). The proof is based on a general construction to simulate a space-bounded Turing machine in an automaton group: similar

[43] In fact, the typical example of a non-finitely presented group, the lamplighter group, is an automaton group [GŻ01]. On the other hand, the typical example of an automaton group, the Grigorchuk group, is not finitely presented [Gri99].

[44] The other two are the conjugacy problem ("are two given group elements conjugated?") and the isomorphism problem ("are two given groups isomorphic?"). Both of them are undecidable for automaton groups [ŠV12].

[45] for example, using automaton minimization techniques (see e. g. [KMP12, subsection 2.3]) or by testing the action on all words up to a certain length (see e. g. [Cai09, Proposition 3.4])

to Kozen's proof that the language intersection emptiness problem for deterministic finite acceptors[46] is PSpace-complete [Koz77, Lemma 3.2.3], the input word for the automaton is considered to encode a computation of a Turing machine and we have various states which check that individual aspects of the computation are valid. The construction originates in the proof that there is an inverse automaton semigroup whose word problem is PSpace-complete [1, Proposition 6] and its main idea is to use binary counters (in a way similar to the adding machine from Example 0.2.1.4) to implement a generalized checkmarking approach. For propagating the information on whether a check of the computation passed or failed, the original construction (for the inverse semigroup case) relied on the fact that the automaton was allowed to be partial.[47] The already existing construction for the inverse semigroup case made it significantly simpler to extend the result to groups. Only the way how the pass or fail information is propagated had to be changed [5, Theorem 10]. The modified construction (which is what we present here) uses iterated commutators to simulate a logical conjunction in the group (compare to [Bar89]).

This general way of simulating Turing machines in automaton groups seems to be quite versatile and, in the last subsection, we apply it to another problem: the compressed word problem,[48] where the input state sequence(s) are not given as words directly but are generated using a context-free grammar. We show that there is an automaton group whose compressed word problem is ExpSpace-complete. In fact, we even show that there is an automaton group whose (normal) word problem is PSpace-complete and whose compressed word problem is ExpSpace-complete, which yields an example of a group whose compressed word problem is provably harder (not only under common assumptions from complexity theory) than its (normal) word problem.

Attribution. As already mentioned, the "guess and check" algorithm was first put into the literature by Steinberg [Ste15, section 3]. This algorithm was analyzed and generalized to (partial) automaton semigroups in joint work with Daniele D'Angeli and Emanuele Rodaro [1]. In fact, the mentioned work even considers the word problem with rational constraints where the question is whether the partial actions of the two given state sequences coincide on all words of a given regular language. The description of the algorithm and the analysis below are based on this work. The lower bound construction in the context of Cain's problem [Cai09, Open problem 3.6] also originates there and the same is true for the generalized checkmarking approach and most of the construction used in the PSpace-hardness proof.

The extension of the PSpace-hardness proof to the group case and the results on the ExpSpace-hardness of the compressed word problem are joint work with Armin Weiß [5]. Some parts of the explanation and proofs (for the PSpace-result) are taken verbatim from this work. The idea of using commutators to simulate logical conjunctions in groups was used by Barrington [Bar89] but similar ideas predate him and have been attributed to Gurevich (see [Mak85]) and given by Mal'cev [Mal62]. That the compressed word problem for automaton structures is in ExpSpace simply follows by uncompressing the straight-line program.

[46] Remember that an acceptor is an automaton without output.

[47] This was one of the main motivations to introduce partial automaton structures in the first place.

[48] The compressed word problem of a group can for example be used to solve the word problem of its automorphism group (if both groups are finitely generated). More on the compressed word problem can be found in [Loh14].

Other results on the word problem of automaton structures (which we do not discuss here) include the result that the word problem of (so-called) contracting automaton groups can be solved in deterministic logarithmic space [Nek05, proof of Proposition 2.13.10]. In particular, this, thus, also holds for automaton groups of bounded activity [Nek05, Theorem 3.9.12] (which we will encounter again in Subsection 2.4.3). The word problem of automaton groups of polynomial activity can be solved in sub-exponential time [Bon12, Corollary 1] and, for Honoi tower groups, one can use the automaton presentation to obtain an algorithm for the word problem running in sub-exponential time as well [Bon14].[49]

[49] In fact, both algorithms seem to be deterministic poly-logarithmic space algorithms.

For the compressed word problem, there does not seem to exist much prior or related work (although some related results can be found in [Bar+20]). However, it was observed by Gillibert (via personal communication) that the proof of [1, Proposition 6] also shows that there is an inverse automaton semigroup with an ExpSpace-hard compressed word problem. This fact came up in a discussion that his construction for an automaton group with an undecidable order problem [Gil18] can be used to prove that there is an automaton group with a PSpace-hard compressed word problem.

2.1.1 A Straight-Forward "Guess and Check" Algorithm

The (Uniform) Word Problem for Automaton Structures. The *uniform word problem for automaton monoids* is the decision problem

Input: an $\mathscr{S}$-automaton $\mathcal{T} = (Q, \Sigma, \delta)$ and two state sequence $\boldsymbol{p}, \boldsymbol{q} \in Q^*$
Question: is $\boldsymbol{p} = \boldsymbol{q}$ in $\mathscr{M}(\mathcal{T})$?

If we limit the two input state sequences to elements of Q^+, we obtain the *uniform word problem for automaton semigroups*.[50] If we alternatively/additionally restrict the input automaton to an $\overline{\mathscr{S}}$-automaton and allow the state sequences to be from $\widetilde{Q}^*$ (or $\widetilde{Q}^+$, respectively), we obtain the *uniform word problem for inverse automaton monoids/semigroups*. The *uniform word problem for automaton groups* is usually stated with only one input state sequence:

[50] Strictly speaking, we also have to ask whether the two state sequences are the same element in $\mathscr{S}(\mathcal{T})$ instead. However, this is the case if and only if they are the same element in $\mathscr{M}(\mathcal{T})$.

Input: a $\mathscr{G}$-automaton $\mathcal{T} = (Q, \Sigma, \delta)$ and a state sequence $\boldsymbol{q} \in \widetilde{Q}^*$
Question: is $\boldsymbol{q} = \mathbb{1}$ in $\mathscr{G}(\mathcal{T})$?

The idea here is that we have $g = h$ if and only if $gh^{-1} = \mathbb{1}$.

Furthermore, we can also remove the automaton from the input altogether and consider the problem for a fixed $\mathscr{S}$-automaton $\mathcal{T}$. This way, we obtain the *word problem* of (the automaton monoid) $\mathscr{M}(\mathcal{T})$:

Constant: an $\mathscr{S}$-automaton $\mathcal{T} = (Q, \Sigma, \delta)$
Input: two state sequences $\boldsymbol{p}, \boldsymbol{q} \in Q^*$
Question: is $\boldsymbol{p} = \boldsymbol{q}$ in $\mathscr{M}(\mathcal{T})$?

Again, we also have versions where the state sequences must be from Q^+ (the word problem of an automaton semigroup) and where the constant automaton is an $\overline{\mathscr{S}}$-automaton and the state sequences may contain inverses (the word problem of an inverse

automaton monoid/semigroup). For the word problem of an automaton group, the constant automaton $\mathcal{T}$ is a $\mathscr{G}$-automaton and we only have a single input state sequence over $\widetilde{Q}$, for which we want to check whether it is the neutral element in the group.

As an upper bound for the complexity of the (uniform) word problem for automaton structures, we can give a rather straight-forward nondeterministic "guess and check" algorithm to solve the problem(s).[51] We will state this algorithm explicitly and analyze it in the next theorem.

[51] As already mentioned, this idea was put into the literature by Steinberg for automaton groups [Ste15, section 3].

Theorem 2.1.1.1 (simplified version of [1, Proposition 2], compare to [Ste15, section 3])**.** *The uniform word problem for automaton monoids*

Input: *an $\mathscr{S}$-automaton $\mathcal{T} = (Q, \Sigma, \delta)$ and two state sequence $\boldsymbol{p}, \boldsymbol{q} \in Q^*$*
Question: *is $\boldsymbol{p} = \boldsymbol{q}$ in $\mathscr{M}(\mathcal{T})$?*

is in $\mathrm{NSPACE}((|\boldsymbol{p}| + |\boldsymbol{q}|) \log |Q| + \log |\Sigma|)$ *and, thus, in linear nondeterministic space.*

Proof. Actually, we give a space-bounded nondeterministic algorithm for the complement problem. By the closure of nondeterministic space classes under complement (see, e. g. [Pap94, Theorem 7.6]), this shows the statement.

An algorithm in pseudo-code can be found in Algorithm 1. The idea is to guess a (shortest) witness $u = a_1 \dots a_\ell \in \Sigma^+$ for $a_1, \dots, a_\ell \in \Sigma$ with $\boldsymbol{p} \circ u \neq \boldsymbol{q} \circ u$ letter by letter. The special handling for the case that one partial action is defined while the other one is not can be found in Algorithm 1 but the general idea is that we have a cross diagram

$$\boldsymbol{p} = \boldsymbol{p}_0 \overset{a_1}{\underset{b_1}{\dashrightarrow}} \boldsymbol{p}_1 \overset{a_2}{\underset{b_2}{\dashrightarrow}} \dots \overset{a_\ell}{\underset{b_\ell}{\dashrightarrow}} \boldsymbol{p}_\ell$$

for $\boldsymbol{p}_1, \dots, \boldsymbol{p}_\ell \in Q^{|\boldsymbol{p}|}$ and $b_1, \dots, b_\ell \in \Sigma$ and a cross diagram

$$\boldsymbol{q} = \boldsymbol{q}_0 \overset{a_1}{\underset{c_1}{\dashrightarrow}} \boldsymbol{q}_1 \overset{a_2}{\underset{c_2}{\dashrightarrow}} \dots \overset{a_\ell}{\underset{c_\ell}{\dashrightarrow}} \boldsymbol{q}_\ell$$

for $\boldsymbol{q}_1, \dots, \boldsymbol{q}_\ell \in Q^{|\boldsymbol{q}|}$ and $c_1, \dots, c_\ell \in \Sigma$. In the algorithm, we guess the letters a_t from left to right until we have found some t with $b_t \neq c_t$. In addition to the current value of a_t (which requires space $\log |\Sigma|$), we only have to store the current value of $\boldsymbol{p}_t$ (requiring space $|\boldsymbol{p}| \log |Q|$) and $\boldsymbol{q}_t$ (requiring space $|\boldsymbol{q}| \log |Q|$).

Clearly, the sequence of guessed a_t yields a witness for the inequality of $\boldsymbol{p}$ and $\boldsymbol{q}$ in $\mathscr{M}(\mathcal{T})$ and, on the other hand, if the two state sequence are unequal in the monoid, there is some witness, which we will guess on some computational branch. □

The uniform word problem for automaton monoids is the most general variant[52] of the word problem in our setting. Thus, as a consequence of Theorem 2.1.1.1, we also get that all other versions of the word problem (where the automaton is constant, invertible

[52] In fact, there is a more general version of the word problem where the partial actions of the two given state sequence do not need to coincide on all words from Σ^* but only on the words of a given regular language. This problem can be solved very similarly and was analyzed in [1, Proposition 2].

Algorithm 1 A nondeterministic algorithm for the (complement of the) uniform word problem for automaton monoids in linear space (simplified version of [1, Algorithm 1]).

```
1  function coWordProblem(𝒯 = (Q, Σ, δ), p, q ∈ Q*): 𝔹;
2  var
3    a ∈ Σ;
4  begin
5    while true do
6      a ← guess(Σ);
7
8      if p ∘ a is defined and q ∘ a is defined then
9        if p ∘ a ≠ q ∘ a then
10         return "p ≠ q in 𝒜(𝒯)";        ▷ The guessed values for a form a witness for the inequality.
11       fi;
12       p ← p · a;
13       q ← q · a;
14     else if (p ∘ a is defined and q ∘ a is undefined) or
15             (p ∘ a is undefined and q ∘ a is defined) then
16       return "p ≠ q in 𝒜(𝒯)";          ▷ One partial action is defined but the other one is not.
17     else                                                ▷ Both partial actions are undefined.
18       fail;                                             ▷ Cancel this computational branch.
19     fi;
20   od;
21 end;
```

and/or complete and we consider the generated (inverse) semigroup or group) are in linear nondeterministic space.

Often,[53] the word problem is solved in a different way: one proves an upper bound on the length of a shortest witness for the inequality of the two state sequences and then checks whether the (partial) actions of the two state sequences coincide on all words up to this length.[54] Such an upper bound also follows from the number of possible configurations of the presented nondeterministic algorithm. The witness for the inequality of the two state sequences $\boldsymbol{p}$ and $\boldsymbol{q}$ in the monoid is given by the sequence of the letters $a_1, \ldots, a_\ell \in \Sigma$ guessed at Line 6 on a successful computational branch. If such a successful configuration can be reached, it can also be reached on a computational branch where the configurations at Line 6 are pairwise distinct (otherwise, we can simply remove the computational loop). Since such a configuration basically consists of the values of $\boldsymbol{p}_t$ and $\boldsymbol{q}_t$ for the current time step t (the value of a is overwritten anyway), there are at most $|Q|^{|\boldsymbol{p}|+|\boldsymbol{q}|}$ many different configurations, which yields the following upper bound.

Corollary 2.1.1.2 (extension of [Cai09, Corollary 3.5] to partial automata, compare to [4, Lemma 5.1])**.** *Let $\mathcal{T} = (Q, \Sigma, \delta)$ be an $\mathscr{S}$-automaton and let $\boldsymbol{p}, \boldsymbol{q} \in Q^*$. We have:*

$$\boldsymbol{p} \neq \boldsymbol{q} \text{ in } \mathscr{M}(\mathcal{T}) \implies \exists u \in \Sigma^* : |u| \leq |Q|^{|\boldsymbol{p}|+|\boldsymbol{q}|} \text{ and } \boldsymbol{p} \circ u \neq \boldsymbol{q} \circ u$$

(including the case that one is defined while the other one is undefined).

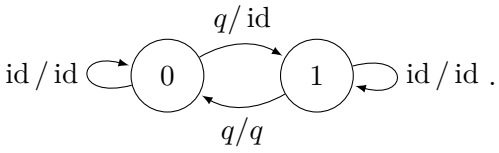
Naturally, this raises the question whether the upper bound can be improved (see [Cai09, Open problem 3.6]). We will shortly see that all versions of the word problem discussed here are PSpace-hard. This shows that there cannot be a an upper bound polynomial in $|\boldsymbol{p}| + |\boldsymbol{q}|$ (unless NP = PSpace) since such an upper bound yields a

[53] see, e. g. [Cai09, Proposition 3.4] (for complete automaton semigroups) but also [Bon12, Corollary 1] or [Bon14]

[54] Alternatively, the problem can also be solved using a (deterministic) minimization process of $\mathcal{T}^{|\boldsymbol{p}|} \uplus \mathcal{T}^{|\boldsymbol{q}|}$. This approach can, for example, be found in [KMP12, subsection 2.3] (for automaton groups).

nondeterministic polynomial time algorithm (for the complement of the word problem): first guess a witness of polynomial length, then check that the partial actions of $\boldsymbol{p}$ and $\boldsymbol{q}$ differ on the guessed word (which can certainly be done in polynomial time). However, we can also explicitly give an exponential lower bound for the length of a shortest witness (for a complete and reversible $\mathscr{S}$-automaton), which we will do in the next example.

Example 2.1.1.3 (see [1, Proposition 4])**.** Recall the adding machine $\mathcal{T} = (Q, \Sigma, \delta)$ from Example 0.2.1.4 and its dual $\partial\mathcal{T}$



For all $n > 0$, we have the cross diagram

$$\begin{array}{ccccc} & & 0^n & & 0 \\ q^{2^n-1} & \longrightarrow & \mathrm{id}^{2^n-1} & \longrightarrow & \mathrm{id}^{2^n-1} \\ & & 1^n & & 0 \\ q & \longrightarrow & q & \longrightarrow & \mathrm{id} \\ & & 0^n & & 1 \end{array}$$

for the adding machine, which shows

$$\boldsymbol{p} \cdot 0^n = \mathrm{id}^{|\boldsymbol{p}|} = \boldsymbol{p} \cdot 0^{n+1}$$

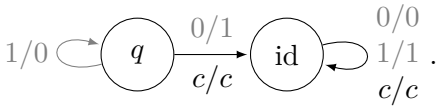
for all $\boldsymbol{p} \in Q^*$ with $|\boldsymbol{p}| < 2^n$ or, in notation for the dual,

$$0^n \circ_\partial \boldsymbol{p} = \mathrm{id}^{|\boldsymbol{p}|} = 0^{n+1} \circ_\partial \boldsymbol{p}.$$

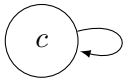
However, it also shows that 0^n and 0^{n+1} are different elements in $\mathscr{S}(\partial\mathcal{T})$. Thus, we have two different state sequences of the dual with lengths n and $n+1$ whose actions coincides on all words of length smaller than

$$2^n = |\Sigma|^{\frac{|0^n|+|0^{n+1}|-1}{2}}.$$

For a different lower bound, we can extend the adding machine into the $\mathscr{S}$-automaton $\mathcal{T} = (Q, \Sigma', \delta')$ by adding the transitions



In the dual, this means adding a new state c and the loops



We obtain $0^n \circ_\partial \boldsymbol{p} = \mathrm{id}^{|\boldsymbol{p}|} = c \circ_\partial \boldsymbol{p}$ for all $\boldsymbol{p} \in Q^*$ with $|\boldsymbol{p}| < 2^n$ but 0^n and c are certainly different elements in $\mathscr{S}(\partial\mathcal{T})$. Thus, we have two state sequences of the dual with lengths n and 1 whose actions coincides on all words of length smaller than

$$2^n = (|\Sigma'| - 1)^{|0^n|+|c|-1}.$$

2.1.2 Simulating a Turing Machine

In this subsection, we will describe a construction for simulating a space-bounded Turing machine in an automaton group. This will allow us to show that there is an automaton group with a PSPACE-complete word problem (proving a conjecture by Steinberg [Ste15, Question 5], see also [AIM07, section 2, 6.])).

A Polynomially Space-Bounded Turing Machine. We fix a deterministic, space-bounded Turing machine M with input alphabet Λ, tape alphabet Δ, blank symbol $\square$, state set P, initial state p_0 and accepting states $F \subseteq P$ and we let $\Gamma = P \uplus \Delta$. The configurations of M are of the form $\square\Delta^\ell P\Delta^m\square$ where $\ell + 1 + m = s(n)$ for the space bound s of M.

For our later reduction, we assume that the space bound s is a polynomial and that the accepted language of M is PSPACE-complete.[55] This means that the word problem of M

Constant:	the machine M
Input:	$w \in \Lambda^*$ of length n
Question:	does M reach a configuration with a state in F from the initial configuration $\square p_0 w \square^{s(n)-n-1}\square$?

is PSPACE-complete and we will use this problem for the reduction[56]: we will construct a $\mathscr{G}$-automaton $\mathcal{T} = (Q, \Sigma, \delta)$ from M (which does not depend on w!) such that we can compute a state sequence from w in LOGSPACE which acts as the identity if and only if M does **not** accept w.

Without loss of generality, we may assume that M satisfies the following normalization property.

[55] Alternatively, we could also take M as a universal Turing machine that can simulate any polynomially space-bounded Turing machine within polynomial space.

[56] In fact, we will rather do a co-reduction!

Fact 2.1.2.1 (Folklore, see [5, Fact 1])**.** *Without altering the accepted language, we may assume that the symbol $\gamma_i^{(t+1)} \in \Gamma$ at position i of a configuration of M at time step $t+1$ only depends on the symbols $\gamma_{i-1}^{(t)}, \gamma_i^{(t)}, \gamma_{i+1}^{(t)} \in \Gamma$ at position $i-1$, i and $i+1$ at time step t. Thus, we may always assume that there is a function $\tau : \Gamma^3 \to \Gamma$ mapping the symbols $\gamma_{i-1}^{(t)}, \gamma_i^{(t)}, \gamma_{i+1}^{(t)} \in \Gamma$ to the uniquely determined symbol $\gamma_i^{(t+1)}$ for all i and t.*

Proof idea (of [5, Fact 1]). The only problem appears if the machine moves to the left: if we have the situation $abpc$ or $abpd$ and the machine moves to the left in state p when reading a c but does not move when reading a d, then the new value for the second symbol does not only depend on the symbols right next to it; we can either be in the situation $ap'bc'$ or $abp'd'$. To circumvent the problem, we can introduce intermediate states. Now, instead of moving to the left, we go into an intermediate state (without movement). In the next step, we move to the left (but this time the movement only depends on the

state and not on the current symbol). Clearly, introducing these intermediate states does not change the behavior of M significantly; in particular, it does not change the accepted language. □

In fact, we will construct $\mathcal{T}$ from τ. The general idea of our construction is similar to the one used by Kozen to show that the language intersection emptiness problem for deterministic finite acceptors[57] is PSPACE-complete [Koz77, Lemma 3.2.3]: we basically consider sequences of configurations of M separated by a special symbol #

$$\gamma_1^{(0)}\gamma_2^{(0)}\gamma_3^{(0)}\dots\gamma_{s(n)}^{(0)}\#\gamma_1^{(1)}\gamma_2^{(1)}\gamma_3^{(1)}\dots\gamma_{s(n)}^{(1)}\#\dots$$

as input words for the automaton and want to check whether the given sequence forms a valid computation of M. We will have special states that perform these checks and store the information whether the check passed or failed. Upon reading another special letter \$, we will switch into a different operational mode of the automaton (which we will call the "A_5-mode"). In this mode (which we will describe below), we will extract the pass or fail information for the checks stored in the states. Concretely, we will have an identity state id as the "fail state" and a state, which we call σ for reasons that will become apparent later, as an "okay state" (see Figure 2.5).

[57] Again, the term *acceptor* refers to an automaton without output.

The most important part of the checks in the first mode of the automaton (the "TM-mode") is to check whether all transitions in the configuration sequence are valid. In Kozen's proof, this is done by having a checking state for every position i with $1 \leq i \leq s(n)$. The acceptor than counts modulo $s(n)$ and locally checks the transition from $\gamma_{i-1}^{(t)}\gamma_i^{(t)}\gamma_{i+1}^{(t)}$ to $\gamma_i^{(t+1)}$ for all t using τ. In our setting, this is not possible, however, since the automaton may only depend on M (or τ, respectively) but not on the length n of the input w.

Generalized Checkmarking. A simple idea to circumvent this problem is to introduce a checkmarking approach. This is possible because the transition table τ does not depend on the position i. The idea of this checkmarking approach is depicted in Figure 2.1. We want to check the transitions at all the first positions without a checkmark (for every time step). For this, we first ignore all positions with a checkmark, perform the check on the first uncheckmarked position and add a checkmark to it (in fact, we want to always add a checkmark independently of whether the check passed or failed); then, we wait for the symbol # and proceed to the next time step. For technical reasons, we split the checkmarking and the actual checking of the symbol into two separate steps.

The problem with this approach is that ignoring all already checkmarked symbols and adding a checkmark to the next position yields an intrinsically non-invertible automaton (as depicted in Figure 2.2). Therefore, we will use a generalized checkmarking approach developed in [1]: we equip every symbol of the configurations with a binary digit block. If the digit block contains only 0s, we consider the symbol as uncheckmarked; if there is at least one 1, we consider it as checkmarked. Now, adding a checkmark to a symbol can be done by incrementing the binary number (and we have already seen how to do this with the adding machine from Example 0.2.1.4). The advantage of this approach is

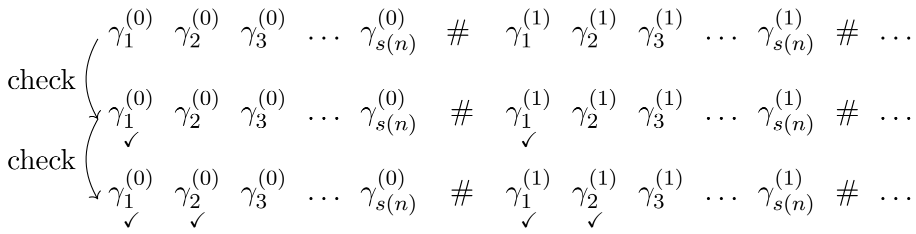


Figure 2.1: Illustration of the checkmark approach ([5, Figure 2])

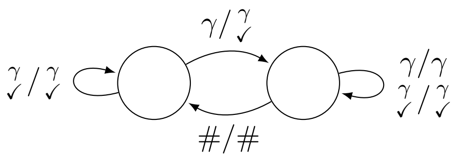


Figure 2.2: Adding a checkmark yields a non-invertible automaton ([5, Figure 3])

that we can increment the binary number again and the corresponding symbol is still considered as checkmarked. Since the digit block is of finite length, we obviously have the problem of overflows but those can be recognized in the automaton and we can add a suitable additional check for this problem (we will go into details in the proof below).

For the construction, it turns out to be a bit simpler if we have the digit block in front of the actual symbol, which leads to the idea for the generalized checkmarking approach depicted in Figure 2.3.

The construction for adding a checkmark to the first so-far uncheckmarked position of every configuration can be found in Figure 2.4. It acts over the alphabet $\Sigma = \Gamma \uplus \{0, 1\} \uplus \{\#, \$\}$ and we make the convention that, whenever a transition with input $x \in \Sigma$ is missing in a state p, there is an implicit $p \xrightarrow{x/x}$ id transition to the identity state id. Additionally, an arrow $p \xrightarrow{\mathrm{id}_X} q$ for a subset $X \subseteq \Sigma$ indicates that we have a transition $p \xrightarrow{x/x} q$ for all $x \in X$. The state g in Figure 2.4 is a placeholder. Later on, we will instantiate the construction with $g = \sigma$, our "okay state", and with $g = \mathrm{id}$, the identity state.

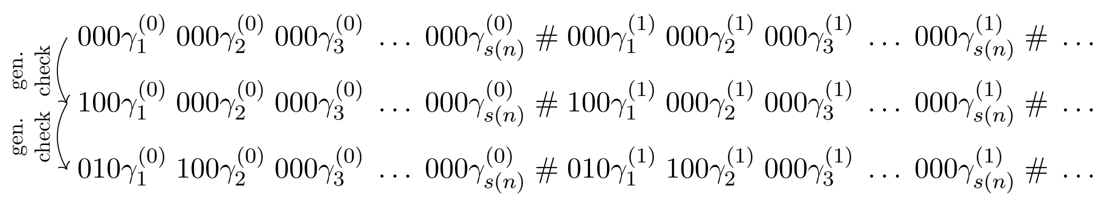


Figure 2.3: The idea of our generalized checkmarking approach ([5, Figure 4])

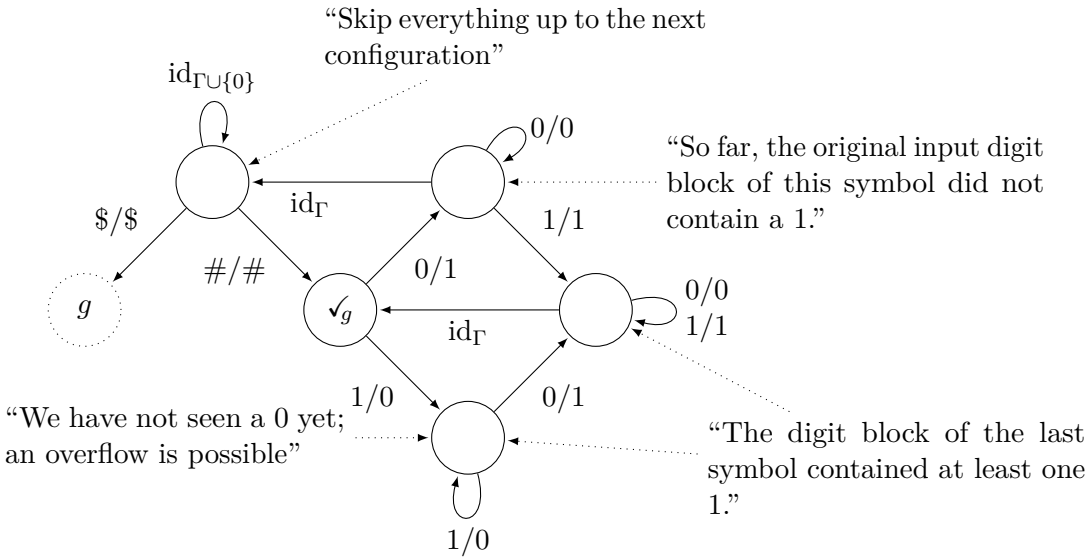


Figure 2.4: The automaton part used for generalized checkmarking (see [5, Figure 5], compare to [1, Fig. 6]); missing transitions go into an identity state (which is not drawn here)

To understand the construction, we will look at the example word

$$100\gamma_1^{(0)}\ 000\gamma_2^{(0)}\ 000\gamma_3^{(0)}\ \#\ 100\gamma_1^{(1)}\ 000\gamma_2^{(1)}\ 000\gamma_3^{(1)}\ \$$$

where the $\gamma_i^{(t)}$ are from Γ [5, p. 6:11]. If we start reading the input word in state $\checkmark_g$, we turn the first 1 into a 0, go to the state at the bottom, turn the next 0 into a 1 and go to the state on the right, where we ignore the next 0. When reading $\gamma_1^{(0)}$, we go back to $\checkmark_g$. Next, we take the upper exit and turn the next 0 into a 1. The remaining 0s are ignored and we remain in the state at the top right until we read $\gamma_2^{(0)}$ and go to the state at the top left. Here, we ignore everything up to #, which gets us back into $\checkmark_g$. The second part works in the same way with the difference that we go to g at the end since we encounter \$ instead of #. The output word, thus, is

$$010\gamma_1^{(0)}\ 100\gamma_2^{(0)}\ 000\gamma_3^{(0)}\ \#\ 010\gamma_1^{(1)}\ 100\gamma_2^{(1)}\ 000\gamma_3^{(1)}\ \$$$

and we have checkmarked the next position in both configurations. In addition, we are in state g after reading \$.

If we look at the word

$$11\gamma_1^{(0)}\ 01\gamma_2^{(0)}\#11\gamma_1^{(1)}\ 01\gamma_2^{(1)}\ \$$$

instead and apply state $\checkmark_g$ to it, we turn the first 1 into a 0 and go to the state at the bottom. Here, we also turn the next 1 into a 0 but we cannot continue with $\gamma_1^{(0)}$. By the above convention, this means that we go into the identity state id, which does not change the rest of the word. In particular, we are also in the "fail state" id after reading \$ (regardless of the value of g). This is how we will eventually detect a counter overflow.

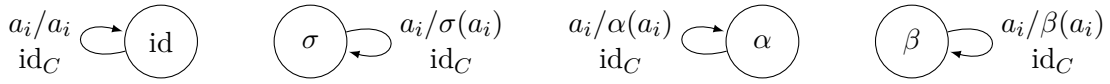


Figure 2.5: States corresponding to the identity and the elements from Fact 2.1.2.2 ([5, p. 6:8])

The Group A_5 and Balanced Iterated Commutators. All our checks will be of a form similar to the one we just described: after reading \$, we are either in the "okay state" σ (if everything succeeded) or we are in the "fail state" id. We now need a way to extract this pass or fail information from the states. In the construction of [1, Proposition 6], this was basically done by omitting a transition if the check failed, which leads to an inverse automaton semigroup. Here, we want to construct a group instead and, thus, cannot apply this approach.

Instead, we will use an idea already applied by Barrington to simulate logical circuits of logarithmic depth and bounded fan-in[58] in a group [Bar89].[59] Just like Barrington, we will use the group A_5 of even permutations over five elements. The reason for this is that it is not solvable and, thus, admits arbitrarily deeply nested non-trivial commutators. The *commutator* of two elements g and h of a group is

$$[h, g] = h^{-1}g^{-1}hg \text{ and}$$

the *conjugate* of g with h (which we also need) is the element

$$g^h = h^{-1}gh.$$

With this notation, we can state the following fact about A_5 (which is stronger than the statement that A_5 is not solvable).

Fact 2.1.2.2 ([5, Lemma 2], see [Bar89, Lemma 1 and 3])**.** *There are $\sigma, \alpha, \beta \in A_5$ with $\sigma \neq \mathbb{1}$ and $\sigma = [\sigma^\beta, \sigma^\alpha]$.*

Proof. Set $\sigma = (13254)$, $\alpha = (23)(45)$ and $\beta = (245)$, for example. □

In the remainder of this subsection, we will make heavy use of these elements. We repartition the alphabet $\Sigma = \Gamma \uplus \{0, 1\} \uplus \{\#, \$\}$ into two disjoint parts $\Sigma = \{a_1, \ldots, a_5\} \uplus C$ where $a_1, \ldots, a_5$ are arbitrary elements and the other elements are contained in C. We may assume that the elements of A_5 are permutations on $a_1, \ldots, a_5$ and we can extend their action first to Σ by letting them act trivially on elements from C and then to Σ^* by applying their action letter-wise. This is what we do for the identity and σ, α, β from Fact 2.1.2.2 with the states depicted from Figure 2.5,[60] which form a part of $\mathcal{T}$. In the following, σ, α and β will usually refer to both the permutations from A_5 and the states from Q.

One exception to this rule is the following definition of a balanced iterated commutator where we use α and β to denote arbitrary[61] state sequences over Q. For this definition, we also extend the notation for commutators and conjugates to state sequences $\boldsymbol{p}, \boldsymbol{q} \in \widetilde{Q}^*$: we let $[\boldsymbol{q}, \boldsymbol{p}]$ denote the state sequence $[\boldsymbol{q}, \boldsymbol{p}] = \overline{\boldsymbol{q}}\,\overline{\boldsymbol{p}}\boldsymbol{q}\boldsymbol{p}$ and $\boldsymbol{p}^{\boldsymbol{q}}$ denotes the state sequence $\boldsymbol{p}^{\boldsymbol{q}} = \overline{\boldsymbol{q}}\boldsymbol{p}\boldsymbol{q}$.

[58] ... so-called NC^1*-circuits*

[59] In fact, similar ideas predate Barrington's result: they have been attributed to Gurevich (see [Mak85]) and given by Mal'cev [Mal62].

[60] Here, we have again used the convention about transitions labeled by id_X from above.

[61] In fact, we will apply the definition later on in such a way that the state sequences indeed correspond to α and β from Fact 2.1.2.2 and Figure 2.5.

Definition 2.1.2.3 ([5, Definition 5])**.** For $\alpha, \beta \in \widetilde{Q}^*$ and $\boldsymbol{p}_d, \dots, \boldsymbol{p}_1 \in \widetilde{Q}^*$, we inductively define the word $B_{\beta,\alpha}[\boldsymbol{p}_d, \dots, \boldsymbol{p}_1]$ by

$$B_{\beta,\alpha}[\boldsymbol{p}_1] = \boldsymbol{p}_1 \quad \text{and}$$
$$B_{\beta,\alpha}[\boldsymbol{p}_d, \dots, \boldsymbol{p}_1] = \left[B_{\beta,\alpha}[\boldsymbol{p}_d, \dots, \boldsymbol{p}_{\lfloor \frac{d}{2} \rfloor + 1}]^\beta,\ B_{\beta,\alpha}[\boldsymbol{p}_{\lfloor \frac{d}{2} \rfloor}, \dots, \boldsymbol{p}_1]^\alpha\right].$$

The idea of the definition is that it allows us to simulate a d-ary logical conjunction in A_5 and, thus, in our automaton group $\mathscr{G}(\mathcal{T})$ if we use the actual states from Figure 2.5 as α and β.

Lemma 2.1.2.4 ([5, Lemma 8])**.** *Let $\boldsymbol{p}_d, \dots, \boldsymbol{p}_1 \in \widetilde{Q}^*$ be state sequences such that, for all $1 \le i \le d$, we have $\boldsymbol{p}_i = \mathrm{id}$ in $\mathscr{G}(\mathcal{T})$ or $\boldsymbol{p}_i = \sigma$ in $\mathscr{G}(\mathcal{T})$. Then, we have*

$$B_{\beta,\alpha}[\boldsymbol{p}_d, \dots, \boldsymbol{p}_1] = \begin{cases} \sigma & \text{if } \boldsymbol{p}_1 = \dots = \boldsymbol{p}_d = \sigma \text{ in } \mathscr{G}(\mathcal{T}) \\ \mathbb{1} & \text{otherwise} \end{cases} \quad \text{in } \mathscr{G}(\mathcal{T}).$$

Proof (of [5, Lemma 8] in the arXiv pre-print). For simplicity, we write B instead of $B_{\beta,\alpha}$ in this proof. For $d = 1$, there is nothing to show. So, let $d > 1$ and, first, assume $\boldsymbol{p}_1 = \dots = \boldsymbol{p}_d = \sigma$ in $\mathscr{G}(\mathcal{T})$. Then, we have

$$B[\boldsymbol{p}_d, \dots, \boldsymbol{p}_1] = [\underbrace{B[\boldsymbol{p}_d, \dots, \boldsymbol{p}_{\lfloor \frac{d}{2} \rfloor + 1}]^\beta}_{=_{\widetilde{\mathcal{T}}}\sigma}, \underbrace{B[\boldsymbol{p}_{\lfloor \frac{d}{2} \rfloor}, \dots, \boldsymbol{p}_1]^\alpha}_{=_{\widetilde{\mathcal{T}}}\sigma}] = \sigma \text{ in } \mathscr{G}(\mathcal{T})$$

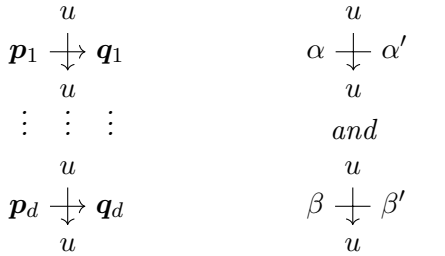
by induction and the choice of σ, α and β from Fact 2.1.2.2. If there is some $i \in \{1, \dots, \lfloor \frac{d}{2} \rfloor\}$ with $\boldsymbol{p}_i = \mathrm{id}$ in $\mathscr{G}(\mathcal{T})$, then, by induction, we have

$$\begin{aligned} B[\boldsymbol{p}_d, \dots, \boldsymbol{p}_1] &= [\underbrace{B[\boldsymbol{p}_d, \dots, \boldsymbol{p}_{\lfloor \frac{d}{2} \rfloor + 1}]^\beta}_{=\boldsymbol{p}'}, \underbrace{B[\boldsymbol{p}_{\lfloor \frac{d}{2} \rfloor}, \dots, \boldsymbol{p}_1]^\alpha}_{=_{\widetilde{\mathcal{T}}}\mathbb{1}}] \\ &= \overline{\beta}\overline{\boldsymbol{p}'}\beta\,\overline{\alpha}\mathbb{1}\alpha\,\overline{\beta}\boldsymbol{p}'\beta\,\overline{\alpha}\mathbb{1}\alpha = \mathbb{1} \text{ in } \mathscr{G}(\mathcal{T}). \end{aligned}$$

The case $i \in \{\lfloor \frac{d}{2} \rfloor + 1, \dots, d\}$ is symmetric. □

Working with a balanced iterated commutator is particularly easy if α, β and all its entries act trivially on the input word. In this case, we can add the commutator to a cross diagram without interfering with the output word. We formulate what this means precisely in the following fact (which can easily be proved using induction).

Fact 2.1.2.5. *Let $u \in \Sigma^*$ and $\alpha, \beta, \boldsymbol{p}_1, \dots, \boldsymbol{p}_d \in \widetilde{Q}^*$ be state sequences such that we have the cross diagrams*



for some $\alpha', \beta', \boldsymbol{q}_1, \ldots, \boldsymbol{q}_d \in \widetilde{Q}^*$*. Then, we also have the cross diagram*

$$B[\boldsymbol{p}_d, \ldots, \boldsymbol{p}_1] \overset{u}{\underset{u}{\dashv\vdash}} B'[\boldsymbol{q}_d, \ldots, \boldsymbol{q}_1]$$

where we write B *for* $B_{\beta,\alpha}$ *and* B' *for* $B_{\beta',\alpha'}$*.*

We will make use of balanced iterated commutators as logical conjunctions to extract the pass or fail information from the states in the A_5-mode of $\mathcal{T}$ (which we enter after reading the first \$). In order to do so, we need to compute the balanced iterated commutator $B_{\beta,\alpha}$ (for arbitrary $\alpha, \beta \in \widetilde{Q}^*$) from the input state sequences $\boldsymbol{p}_d, \ldots, \boldsymbol{p}_1$ in LogSpace, however.

Lemma 2.1.2.6 ([5, Lemma 6])**.** *For any* $\alpha, \beta \in \widetilde{Q}^*$*, one can compute* $B_{\beta,\alpha}[\boldsymbol{p}_d, \ldots, \boldsymbol{p}_1]$ *in logarithmic space on input of* $\boldsymbol{p}_d, \ldots, \boldsymbol{p}_1$*.*

Proof (of [5, Lemma 6] in the arXiv pre-print). We give a sketch for a (deterministic) algorithm which computes the symbol at position i of $B_{\beta,\alpha}[\boldsymbol{p}_d, \ldots, \boldsymbol{p}_1]$ in logarithmic space. For simplicity, we only describe the case when $d = 2^k$ for some k (as we can easily add dummy entries behaving as σ if necessary). Then, we have

$$\begin{aligned} B_{\beta,\alpha}[\boldsymbol{p}_d, \ldots, \boldsymbol{p}_1] = \; & \overline{\beta}\overline{B_{\beta,\alpha}[\boldsymbol{p}_d, \ldots, \boldsymbol{p}_{\frac{d}{2}+1}]}\beta\; \overline{\alpha}\overline{B_{\beta,\alpha}[\boldsymbol{p}_{\frac{d}{2}}, \ldots, \boldsymbol{p}_1]}\alpha \\ & \overline{\beta}B_{\beta,\alpha}[\boldsymbol{p}_d, \ldots, \boldsymbol{p}_{\frac{d}{2}+1}]\beta\; \overline{\alpha}B_{\beta,\alpha}[\boldsymbol{p}_{\frac{d}{2}}, \ldots, \boldsymbol{p}_1]\alpha \end{aligned}$$

and the length $\ell(d)$ (as a word over α, β, the $\boldsymbol{p}_i$ and their inverses) of $B_{\beta,\alpha}[\boldsymbol{p}_d, \ldots, \boldsymbol{p}_1]$ is given by $\ell(1) = 1$ and $\ell(d) = 8 + 4\ell(\frac{d}{2})$. This yields

$$\ell(d) = \left(\sum_{i=0}^{k-1} 4^i \cdot 8\right) + 4^k = 8\frac{4^k - 1}{3} + 4^k = \frac{8}{3}(d^2 - 1) + d^2 = \frac{11}{3}d^2 - \frac{8}{3}$$

and, thus, that the length of $B_{\beta,\alpha}[\boldsymbol{p}_d, \ldots, \boldsymbol{p}_1]$ is polynomial in d (and can be computed in LogSpace). Therefore, we can iterate the above algorithm for all positions $1 \leq i \leq \ell(d)$ to output $B_{\beta,\alpha}[\boldsymbol{p}_d, \ldots, \boldsymbol{p}_1]$ entirely.

To compute the symbol at position i, we first check whether i is the first or last position (notice that we need the exact value of $\ell(d)$ for testing the latter). In this case, we know that it is $\overline{\beta}$ or α, respectively. Similarly, we can do this for the positions in the middle and at one or three quarters. If the position falls into one of the four recursion blocks, we use two pointers into the input: `left` and `right`. Depending on the block, `left` and `right` either point to $\boldsymbol{p}_1$ and $\boldsymbol{p}_{\frac{d}{2}}$ or to $\boldsymbol{p}_{\frac{d}{2}+1}$ and $\boldsymbol{p}_d$. Additionally, we also store whether we are in an inverse block or a non-inverse block. From now on, we disregard the input left of `left` and right of `right` (and do appropriate arithmetic on i) and proceed recursively. If we need to perform another recursive step, we update the variables `left` and `right` (instead of using new ones). Therefore, the whole recursion can be done in logarithmic space. □

Simulating the Machine M. With the generalized checkmarking approach and the balanced iterated commutator, we have introduced the two main ideas for the construction. Therefore, we can now proceed with the main proof.

Theorem 2.1.2.7 ([5, Theorem 10], compare to [1, Proposition 6])**.** *There is an automaton group whose word problem*

Constant: *a $\mathscr{G}$-automaton $\mathcal{T} = (Q, \Sigma, \delta)$*
Input: *a word $\boldsymbol{q} \in \widetilde{Q}^*$*
Question: *is $\boldsymbol{q} = \mathbb{1}$ in $\mathscr{G}(\mathcal{T})$?*

is PSPACE*-complete.*

Proof (contains parts of the proof of [5, Theorem 10]). Theorem 2.1.1.1 implies that the word problem of every automaton group is in PSPACE. Thus, we only have to show the hardness part. As we have already discussed, we will construct the $\mathscr{G}$-automaton $\mathcal{T} = (Q, \Sigma, \delta)$ from M (or τ, rather) and reduce the word problem of M

Constant: the machine M
Input: $w \in \Lambda^*$ of length n
Question: does M reach a configuration with a state in F from the initial configuration $\square p_0 w \square^{s(n)-n-1} \square$?

in logarithmic space to the (complement of the) word problem of $\mathscr{G}(\mathcal{T})$. As the former problem is PSPACE-hard (and as PSPACE is closed under taking the complement), we obtain that the word problem of $\mathscr{G}(\mathcal{T})$ is PSPACE-hard as well.

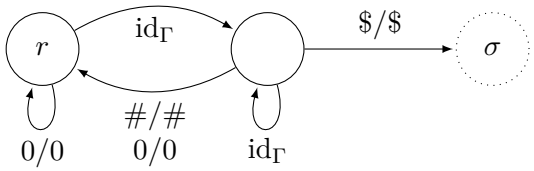
The Definition of $\mathcal{T}$. The automaton $\mathcal{T}$ operates in two phases (the TM-mode and the A_5-mode) and, as its alphabet, we have already chosen $\Sigma = \Gamma \uplus \{0,1\} \uplus \{\#, \$\} = \{a_1, \ldots, a_5\} \uplus C$ (where $\Gamma = P \uplus \Delta$ is the alphabet of the configurations of M). The typical input words for $\mathcal{T}$ will be of the form $u\$v$ with $u \in \Sigma^* \setminus \{\$\}$. While reading u, the automaton will be in TM-mode and, upon reading the first \$, it will switch to the A_5-mode. Accordingly, we call u the "TM-part" and v the "A_5-part".

We have already encountered some of the states of $\mathcal{T}$ in Figure 2.5, where id has the intuition of a "fail state" and σ has the intuition of an "okay state". All these states belong to the A_5-mode of the automaton and we also need states corresponding to α and β that ignore the TM-part of the input word and then go to α and β, respectively (for the A_5-mode) [5, p. 6:9]:

Here, we have introduced the convention that dotted states refer to ones already defined above.[62]

[62] Remember also our conventions about missing transitions going to id and transitions labeled by id_X from above.

The remaining parts of the automaton belong to the TM-mode and most of them check whether the input word is well-formed and describes a valid and accepting computation of the Turing machine M. The first part is responsible for checking that the TM-part of the input word is from $(0^*\Gamma)^+ (\#(0^*\Gamma)^+)^*$ [5, p. 6:9]:

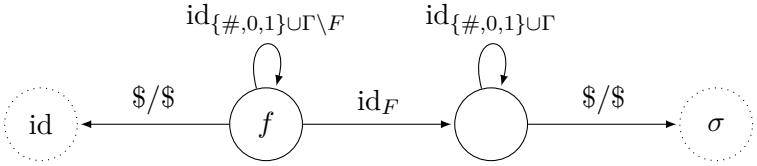


If we start reading an input word in r, then we are in σ after reading the first \$ if the word is of the correct form; otherwise, we end in id. Thus, the TM-part of the input word will typically be of the form

$$0^{\ell_1^{(0)}}\gamma_1^{(0)}\,0^{\ell_2^{(0)}}\gamma_2^{(0)}\;\ldots\;0^{\ell_{I_0}^{(0)}}\gamma_{I_0}^{(0)}\,\#\ldots\#\,0^{\ell_1^{(T)}}\gamma_1^{(T)}\,0^{\ell_2^{(T)}}\gamma_2^{(T)}\;\ldots\;0^{\ell_{I_T}^{(T)}}\gamma_{I_T}^{(T)} \qquad (\dagger)$$

with $\gamma_i^{(t)} \in \Gamma$ and it helps to consider the remaining parts of $\mathcal{T}$ only with respect to such input words.

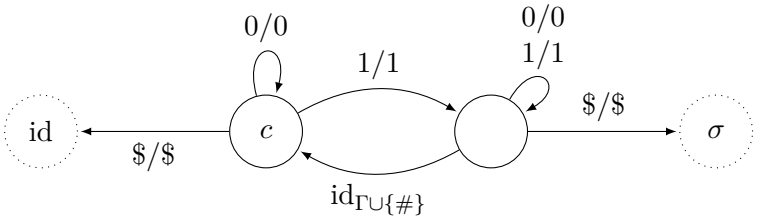
Next, we add a part that verifies that the computation contains an accepting state from F [5, p. 6:9]:



Again, if we start in f, we end in σ if and only if the part before the first \$ contains a symbol from F; otherwise, we end in id.

Before we describe the automaton part which actually verifies that all transitions are valid, we first introduce some parts required for the generalized checkmark approach outlined above. For this, we add the already discussed part from Figure 2.4 once for $g = \sigma$ (this yields the state $\checkmark_\sigma$) and once for $g = \mathrm{id}$ (which yields the state $\checkmark_{\mathrm{id}}$). The state $\checkmark_\sigma$ goes to σ in the A_5-mode if no overflows occurred and the state $\checkmark_{\mathrm{id}}$ always ends in id for the A_5-mode.

In addition, we also add the following part containing the state c, which can be used to verify that all symbols of all configurations haven been checkmarked (in the generalized sense) [5, p. 6:9]:



Finally, we come to the most technical part, which is used for verifying the transitions (see [5, pp. 6:10–11]). Intuitively, for checking the transition from time step $t-1$ to time step t at position i, we need to compute $\gamma_i^{(t)} = \tau(\gamma_{i-1}^{(t-1)}, \gamma_i^{(t-1)}, \gamma_{i+1}^{(t-1)})$ from the configuration symbol at positions $i-1$, i and $i+1$ for time step $t-1$. We store $\gamma_i^{(t)}$ in

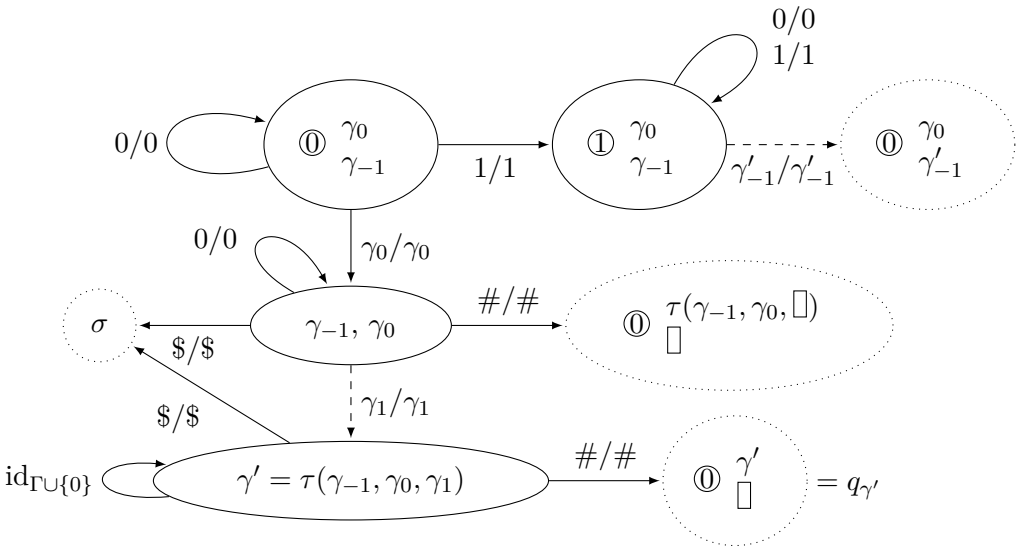
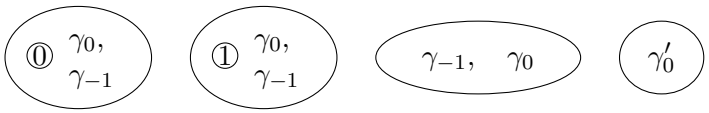


Figure 2.6: Schematic representation of the transitions used for checking Turing machine transitions and definition of q'_γ; the dashed transitions exist for all γ'_{-1} and γ_1 in Γ but go to different states, respectively ([5, Figure 6], compare to [1, Fig. 7])

the state (to compare it to the actual value). Additionally, we need to store the last two symbols we have encountered so far of the configuration at t (for computing what we expect at the next time step later on) and whether we have seen a 1 or only 0s in the checkmark digit block.

For all this, we use the states



with $\gamma_{-1}, \gamma_0, \gamma'_0 \in \Gamma$. The idea is the following. In the Ⓞ and ① states, we store the value we expect for the first unchecked symbol (γ_0) and the last symbol we have seen in the current configuration (γ_{-1}). We are in the Ⓞ-state if we have not seen any 1 in the digit block yet and in the corresponding ①-state if we did. The latter two are used to skip the rest of the current configuration and to compute the symbol we expect for the first uncheckmarked position in the next configuration (γ'_0).

We use these states in the transitions schematically depicted in Figure 2.6. Here, the dashed transitions exist for all γ'_{-1} and γ_1 in Γ but go to different states, respectively, and the dotted states correspond to the respective non-dotted states with different values for γ_0 and γ_{-1} (with the exception of σ, which corresponds to the state from Figure 2.5). We also define $q_{\gamma'}$ as the state on the bottom right (for every $\gamma' \in \Gamma$, respectively).

To understand this automaton part, we make another example. We will use the

example word

$$000\gamma_1^{(0)}\ 000\gamma_2^{(0)}\ 000\gamma_3^{(0)}\ \#\ 000\gamma_1^{(1)}\ 000\gamma_2^{(1)}\ 000\gamma_3^{(1)}\ \$,$$

which is similar to the one used for explaining the generalized checkmarking construction above. Suppose we want to check the transition at position 2 from the first configuration to the second one. We first create a situation where the second positions are the respective first ones that are not checkmarked yet. To achieve this, we first apply the state $\checkmark_{\mathrm{id}}$, which yields the output word

$$100\gamma_1^{(0)}\ 000\gamma_2^{(0)}\ 000\gamma_3^{(0)}\ \#\ 100\gamma_1^{(1)}\ 000\gamma_2^{(1)}\ 000\gamma_3^{(1)}\ \$.$$

We know the initial configuration for M on input w (it is $\square p_0 w \square^{s(n)-n-1}\square$) and, thus, the symbols that we expect for $\gamma_i^{(0)}$. For the example, we assume that the first configuration in the example word matches this expectations. Therefore, we expect the symbol $\gamma_2^{(0)}$ at the second position of the first configuration and we use the state $q_{\gamma_2^{(0)}}$ to encode this. If we apply this state to the output word above, we start in the state at the top left in Figure 2.6 (with $\gamma_0 = \gamma_2^{(0)}$ and $\gamma_{-1} = \square$) and immediately take the transition labeled with 1/1 to the corresponding ①-state, where we ignore the remaining two digits. Upon reading $\gamma_1^{(0)}$, we go to the state at the top right, which is effectively the same as the state on the top left (with $\gamma_0 = \gamma_2^{(0)}$ and $\gamma_{-1} = \gamma_1^{(0)}$, this time). There, we ignore the next three 0s and go to the state in the middle using the $\gamma_2^{(0)}/\gamma_2^{(0)}$-transition. Note that, here, it is important that the actual symbol in the configuration matches our expectation! If this was not the case, we could have only used a transition to the identity state. Next, we ignore the next digit block and go to the state at the bottom left by reading $\gamma_3^{(0)}$. The configuration symbol stored in this state is $\gamma' = \tau(\gamma_1^{(0)}, \gamma_2^{(0)}, \gamma_3^{(0)})$ and we finally go to the state $q_{\gamma'}$ by reading #.

This means that we start again at the top left state and read the second configuration is a similar way. Here, the interesting part is before we take the γ_0/γ_0-transition because we can have two cases: either $\gamma_2^{(1)}$ matches γ' or it does not. If it does, we continue just like before and finally go to σ when reading the final \$. If it does not, we go to the identity state, in which we still have to be after reading the \$. In this way, we have implemented a check for the second positions of all configurations where we either end in σ (if the check passed) or in id (if the check failed) for the A_5-mode.

This concludes the definition of $\mathcal{T}$ and it remains to describe the state sequence $\boldsymbol{q}$ and how it is computed from w.

Definition of the State Sequence. We will choose the state sequence $\boldsymbol{q}$ as an iterated balanced commutator (as in Definition 2.1.2.3) whose individual entries are state sequences that perform checks for various aspects of the input word. The state sequence $\boldsymbol{q}$ depends on the input word $w \in \Lambda^*$ of length n and needs to be computable in logarithmic space (with respect to n). The idea is that, for a check state sequence $\boldsymbol{p}$ and an input word of the form $u\$v$ with $u \in \Sigma^* \setminus \{\$\}$, $\boldsymbol{p} \cdot u\$$ is either equal to σ (if the check passed) or equal

to id (if it failed) in $\mathscr{G}(\mathcal{T})$. Furthermore, the check state sequences will operate trivially on the TM-part u so that the individual checks do not interfere (as we want to apply Fact 2.1.2.5). Eventually, the idea is to use the commutator as a logical conjunction (as outlined in Lemma 2.1.2.4) to test whether all checks passed.

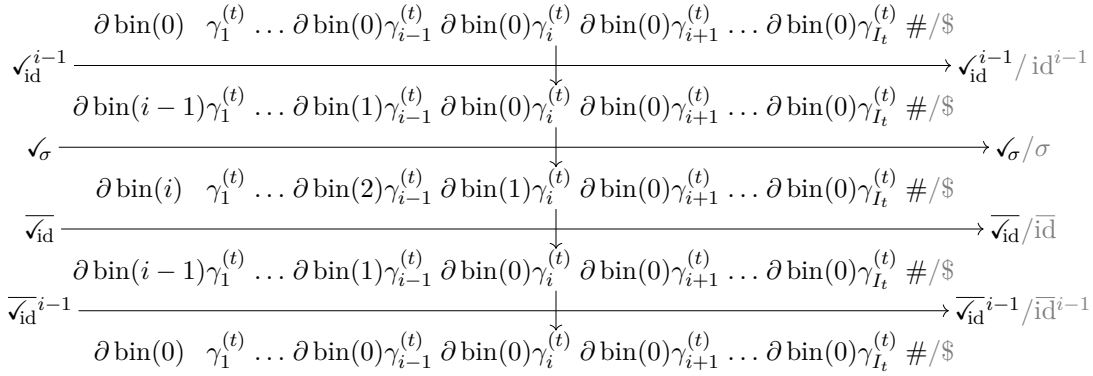
The first check consists of the state r from above. We have already discussed that it will end in σ after reading $u\$$ if and only if u is from $(0^*\Gamma)^+ (\#(0^*\Gamma)^+)^*$. From the construction, we also immediately obtain that it does not change u. Because of this check, we will only need to consider input words whose TM-part is of the form (†) any further.

Next, we verify that all positions $1 \leq i \leq s(n)$ can be checkmarked (in the generalized sense) without any overflows. This is done by the state sequences

$$\boldsymbol{c}_i = \overline{\checkmark_{\mathrm{id}}}^{\,i} \checkmark_\sigma \checkmark_{\mathrm{id}}^{i-1}$$

for $1 \leq i \leq s(n)$. Let $\partial\,\mathrm{bin}(z)$ denote the reverse/least significant bit first binary representation[63] (of a suitable length) for the natural number z. If all digit blocks are long enough, we have the cross diagrams (see [5, p. 6:12])

[63] see Example 0.2.1.4



by the transitions in Figure 2.4. In particular, the TM-part of the input word is not changed.[64] If the digit block belonging to the symbol $\gamma_j^{(t)}$ is too short to count to the value required by $\boldsymbol{c}_i$ (i. e. if an overflow occurs), $\boldsymbol{c}_i \cdot u\$$ will only consist of identity states (and the check is considered to have failed). Additionally, if we have $I_t < i$ (i. e. if one configuration does not have length at least i), we will have the same situation. Thus, using the checks $\boldsymbol{c}_i$ for all $1 \leq i \leq s(n)$, we also ensure that all configurations have length at least $s(n)$.

[64] This is why we count back to zero in the end.

Complementary to this, we also need to check that no configuration is longer than[65] $s(n)$. For this, we use

$$\boldsymbol{c}' = \overline{\checkmark_{\mathrm{id}}}^{\,s(n)} c \checkmark_{\mathrm{id}}^{s(n)},$$

which checks that, after checkmarking the first $s(n)$ many positions in every configuration, all positions are checkmarked.

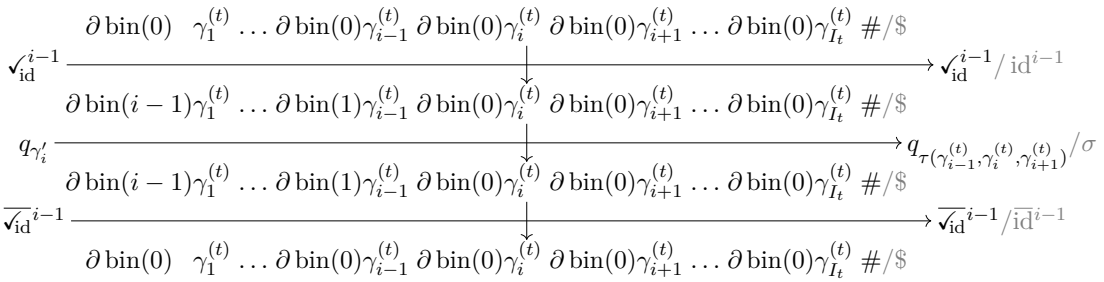
[65] This could cause mistakes when checking the transitions or whether an accepting state from F occurs (in a valid way).

These checks guarantee that the TM-part of the input word is of the from given in (†) and that we have $I_t = s(n)$ for all t. It remains to check that the $\gamma_i^{(t)}$ belong to a valid

computation of M with initial configuration $\square p_0 w \square^{s(n)-n-1}\square$ that reaches an accepting state from F. Let $\gamma'_1 \dots \gamma'_{s(n)} = p_0 w \square^{s(n)-n-1}$ with $\gamma'_i \in \Gamma$ for $1 \leq i \leq s(n)$ and define the state sequences

$$\boldsymbol{q}_i = \overline{\checkmark_{\mathrm{id}}}^{i-1} q_{\gamma'_i} \checkmark_{\mathrm{id}}^{i-1}$$

for $1 \leq i \leq s(n)$. If $\gamma_i^{(t)}$ matches the expected symbol γ'_i (and if no overflows occur during the checkmarking), we obtain the cross diagram (see [5, p. 6:12])



from the transitions in Figure 2.6. By induction on the configurations, we obtain $\boldsymbol{q}_i \cdot u\$ = \sigma$ in $\mathscr{G}(\mathcal{T})$ if all transitions at position i are valid (and, of course, we do not encounter overflows). On the other hand, if $\gamma_i^{(t)}$ does not match the expected symbol γ'_i, we will end in a state sequence containing only identity states in the A_5-mode. Obviously, the TM-part of the input word is also not changed by any $\boldsymbol{q}_i$.

Finally, we use the state f from above to check that there is at least one accepting state from F. By construction, f also does not change the TM-part and ends in σ or id in the A_5-mode.

To combine the individual checks, we use an iterated balanced commutator. We define B_0 as a short-hand notation for B_{β_0,α_0} (as defined in Definition 2.1.2.3) and let

$$\boldsymbol{q} = B_0[f, \boldsymbol{q}_{s(n)}, \dots, \boldsymbol{q}_1, \boldsymbol{c}', \boldsymbol{c}_{s(n)}, \dots, \boldsymbol{c}_1, r].$$

Clearly, the individual checking state sequences can be computed in LogSpace and, by Lemma 2.1.2.6, we can also compute the iterated balanced commutator from them in LogSpace.

Correctness ([5, pp. 6:13–14])**.** We prove that $\boldsymbol{q}$ acts as the identity if and only if the Turing machine M does **not** accept the input word w of length n. The easier direction is to assume that the Turing machine accepts on the initial configuration $\square p_0 w \square^{s(n)-n-1}\square$. Let $\gamma_1^{(0)} \dots \gamma_{s(n)}^{(0)} \vdash \gamma_1^{(1)} \dots \gamma_{s(n)}^{(1)} \vdash \dots \vdash \gamma_1^{(T)} \dots \gamma_{s(n)}^{(T)}$ be the corresponding computation with $\gamma_1^{(0)} = p_0$, $\gamma_2^{(0)} \dots \gamma_{n+1}^{(0)} = w$, $\gamma_{n+2}^{(0)}, \dots, \gamma_{s(n)}^{(0)} = \square$ and $\gamma_i^{(T)} \in F$ for some $1 \leq i \leq s(n)$. We choose $\ell = \lceil \log(s(n)) \rceil + 1$ and let

$$u = 0^\ell \gamma_1^{(0)} \dots 0^\ell \gamma_{s(n)}^{(0)} \# 0^\ell \gamma_1^{(1)} \dots 0^\ell \gamma_{s(n)}^{(1)} \# \dots \# 0^\ell \gamma_1^{(T)} \dots 0^\ell \gamma_{s(n)}^{(T)}.$$

We now let $\boldsymbol{q}$ act on the word $u\$a_1$ where we assume (without loss of generality) that σ acts non-trivially on a_1 (which is possible because σ belongs to a non-trivial permutation

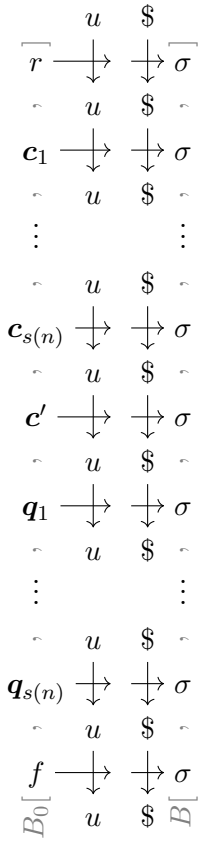


Figure 2.7: Cross diagram for $\boldsymbol{q}$ [5, Figure 7]

from A_5). This yields the black part of the cross diagram depicted in Figure 2.7. From Fact 2.1.2.5, we immediately also obtain the gray additions to the cross diagram where we use B instead of $B_{\beta,\alpha}$ for the balanced commutator from Definition 2.1.2.3. By Lemma 2.1.2.4, we obtain $B[\sigma,\dots,\sigma] = \sigma$ in $\mathscr{G}(\mathcal{T})$. Therefore, $\boldsymbol{q}$ acts non-trivially on $u\$a_1$.

For the other direction, assume that no valid computation of M on the initial configuration $\square p_0 w \square^{s(n)-n-1}\square$ contains an accepting state from F. We have to show that $\boldsymbol{q}$ acts like the identity on all words from Σ^*. If the word does not contain a \$, then all individual parts of $\boldsymbol{q}$ act on it like the identity by construction. This is clearly the case for r, $\boldsymbol{c}'$, the $\boldsymbol{q}_i$ and f. For the $\boldsymbol{c}_i$, the only point to note is that $\checkmark_\sigma$ acts in the same way as $\checkmark_{\mathrm{id}}$ on such words.

Thus, we may assume that the word is of the form $u\$v$. If u is not of the form $(0^*\Gamma)^+\,(\#(0^*\Gamma)^+)^*$, we have the cross diagram

$$\begin{array}{ccc} u & \$ & \\ r & & \mathrm{id} \\ u & \$ & \end{array} \quad \text{and, thus,} \quad \begin{array}{ccc} u & \$ & \\ \boldsymbol{q} & & B[\boldsymbol{p}_d,\dots,\boldsymbol{p}_2,\mathrm{id}] \\ u & \$ & \end{array}$$

for some $\boldsymbol{p}_2,\dots,\boldsymbol{p}_d \in \widetilde{Q}^*$ such that every $\boldsymbol{p}_i$ is equal to σ or equal to id in $\mathscr{G}(\mathcal{T})$. As we have, $B[\boldsymbol{p}_d,\dots,\boldsymbol{p}_2,\mathrm{id}] = \mathbb{1}$ in $\mathscr{G}(\mathcal{T})$ by Lemma 2.1.2.4, we obtain that $\boldsymbol{q}$ acts like the identity on $u\$v$.

Therefore, we assume u to be of the form (†) and use a similar argumentation for the remaining cases. If u does not contain a state from F, then we end up in state id after reading \$ for f. As w is not accepted by the machine, this includes in particular all valid computations on the initial configuration $\square p_0 w \square^{s(n)-n-1}\square$. If one of the 0 blocks in u is too short to count to a value required for the checkmarking (i. e. one $\ell_i^{(t)}$ is too small), then the corresponding $\boldsymbol{c}_i$ will go to (a state sequence equivalent to) id. This is also true if one configuration is too short (i. e. $I_t < s(n)$ for some t). If one configuration is too long (i. e. $I_t > s(n)$), then this will be detected by $\boldsymbol{c}'$ as not all positions will be checkmarked after checkmarking all first $s(n)$ positions in every configuration. Finally, $\boldsymbol{q}_i$ yields an id if $\gamma_i^{(0)}$ is not the correct symbol from the initial configuration or if we have $\gamma_i^{(t+1)} \neq \tau(\gamma_{i-1}^{(t)}, \gamma_i^{(t)}, \gamma_{i+1}^{(t)})$ for some t (where we let $\gamma_{-1}^{(t)} = \square = \gamma_{s(n)+1}^{(t)}$). □

2.1.3 Compressed Word Problem

Straight-Line Programs. A *context-free grammar* (over Σ) is a tuple $\mathcal{G} = (V, \Sigma, P, S)$ where V is an alphabet of *variables*, Σ is an alphabet of *terminal symbols* with $V \cap \Sigma = \emptyset$, $P \subseteq V \times (V \cup \Sigma)^*$ is the set of *production rules* and $S \in V$ is the *start variable*. For such a context-free grammar,[66] we define the relation $\to \subseteq (V \cup \Sigma)^* \times (V \cup \Sigma)^*$ by

$$\lambda A \rho \to \lambda \mu \rho \iff (A, \mu) \in P$$

for all $\lambda, \mu, \rho \in (V \cup \Sigma)^*$ and $A \in V$. Note that we have $A \to \mu$ if and only if $(A, \mu) \in P$. The reflexive and transitive hull of $\to$ is $\to^*$. A variable $A \in V$ is called *productive* if

[66] To which grammar the relation refers will always be obvious from the context.

there is some $w \in \Sigma^*$ with $A \to^* w$ and a production rule $(A, \mu) \in P$ is called *effective* if there are $\lambda, \rho \in (V \cup \Sigma)^*$ and $w \in \Sigma^*$ with $S \to^* \lambda A \rho \to \lambda \mu \rho \to^* w$. A variable $A \in V$ is *effective* if there is an effective production rule $(A, \mu) \in P$.

The *language* generated by a variable $A \in V$ is the set

$$\mathscr{L}(A) = \{w \in \Sigma^* \mid A \to^* w\}$$

and the language generated by $\mathcal{G}$ is $\mathscr{L}(\mathcal{G}) = \mathscr{L}(S)$. If the generated language of a context-free grammar is a singleton set, the grammar is called a *straigt-line program* for the unique word in the generated language, which we denote by $\mathscr{W}(\mathcal{G})$. In the same way, we also write $\mathscr{W}(A)$ for the unique element of $\mathscr{L}(A)$ if we are dealing with a singleton set. Note that this is the case for all effective variables of a straight-line program.

We use the term *degree* of a context-free grammar $\mathcal{G} = (V, \Sigma, P, S)$ here for the maximal length of the second component of a production rule in P, i. e. it is the number $\max\{|\mu| \mid \exists A \in V : A \to \mu\}$. The degree can be used to derive an upper bound on the length of the word generated by a straight-line program.

Fact 2.1.3.1. *Let $\mathcal{G} = (V, \Sigma, P, S)$ be a straight-line program and let $D = \max\{|\mu| \mid \exists A \in V : A \to \mu\}$. Then, we have:*

$$|\mathscr{W}(\mathcal{G})| \leq D^{|V|}$$

Proof. If we have $D = 0$, we must also have $S \to \varepsilon$ and the statement holds. Otherwise, we may assume that all variables of $\mathcal{G}$ are productive and all production rules of $\mathcal{G}$ are effective since it can only decreases the size of V and the value of D if we remove those which are not. This implies that, for every $A \in V$, there is exactly one rule of the form (A, μ) with $\mu \in (V \cup \Sigma)^*$ in P. Additionally, we may assume that all letters from Σ appear in $\mathscr{W}(\mathcal{G})$. Then, the graph given by the node set $V \cup \Sigma$ and the edge set

$$\{A \to X \mid A \in V, X \in V \cup \Sigma \text{ such that } A \to \lambda X \rho \text{ for some } \lambda, \rho \in (V \cup \Sigma)^*\}$$

is a directed tree whose root is S, whose leaves are given by Σ and the variables $A \in V$ with $A \to \varepsilon$ and whose maximal out-degree is D. We can assign a *level* to each node $X \in V \cup \Sigma$ of the tree: it is the maximal number of edges needed to go from X to a leaf, i. e. leaves get level 0, nodes that only have leaves as children get level 1 and, in general, an inner node has level $i + 1$ if and only if it has a child of level i. Note that the level of the root S can at most be $|V|$.

We set $\mathscr{W}(a) = a$ for all $a \in \Sigma$ and show that $\mathscr{W}(X)$ for a node X of level i has length $|\mathscr{W}(X)| \leq D^i$ by induction on i. For the leaves with $i = 0$, we either have $X \in \Sigma$ and, thus, $|\mathscr{W}(X)| = 1 = D^0$ or $X \to \varepsilon$ and, thus, $|\mathscr{W}(X)| = 0 < D^0$. For a node $X \in V$ with level $i > 0$, we have that all children have level smaller than i. Thus, for the unique production rule $(X, Y_1 \dots Y_d)$ with $Y_1, \dots, Y_d \in V \cup \Sigma$ and $0 \leq d \leq D$ in P, we have $|\mathscr{W}(Y_k)| \leq D^{i-1}$ for all $0 \leq k \leq d$ and, thus,

$$|\mathscr{W}(X)| = |\mathscr{W}(Y_1) \dots \mathscr{W}(Y_d)| \leq dD^{i-1} \leq D^i. \qquad \square$$

The Compressed Word Problem. The difference between the (normal) word problem(s) and the compressed word problem(s) is that the state sequences are not given directly but as straight-line programs. Accordingly, the *uniform compressed word problem for automaton monoids* is

Input: an $\mathscr{S}$-automaton $\mathcal{T} = (Q, \Sigma, \delta)$ and
two straight-line programs $\mathcal{P}$ and $\mathcal{R}$ over Q
Question: is $\mathscr{W}(\mathcal{P}) = \mathscr{W}(\mathcal{R})$ in $\mathscr{M}(\mathcal{T})$?

The compressed versions of the other uniform word problems for automaton structures are defined in the obvious way and the *compressed word problem* of an automaton group generated by a $\mathscr{G}$-automaton $\mathcal{T}$ is

Constant: a $\mathscr{G}$-automaton $\mathcal{T} = (Q, \Sigma, \delta)$
Input: a straight-line program $\mathcal{P}$ over $\widetilde{Q}$
Question: is $\mathscr{W}(\mathcal{P}) = \mathbb{1}$ in $\mathscr{G}(\mathcal{T})$?

An obvious way to solve the compressed word problem is to uncompress the input straight-line program(s) and to apply the algorithm for the (normal) word problem afterwards. This way, we immediately obtain the following space bound by combining Fact 2.1.3.1 and Theorem 2.1.1.1.

Corollary 2.1.3.2. *The uniform compressed word problem for automaton monoids*

Input: *an $\mathscr{S}$-automaton $\mathcal{T} = (Q, \Sigma, \delta)$ and*
two straight-line programs $\mathcal{P} = (V, Q, P, S)$ and $\mathcal{R} = (W, Q, R, T)$
Question: *is $\mathscr{W}(\mathcal{P}) = \mathscr{W}(\mathcal{R})$ in $\mathscr{M}(\mathcal{T})$?*

is in NSpace$((D^{|V|} + E^{|W|}) \log |Q| + \log |\Sigma|)$ *and, thus, in* ExpSpace *where* $D = \max\{|\mu| \mid \exists A \in V : A \to \mu \text{ for } \mathcal{P}\}$ *is the degree of $\mathcal{P}$ and E is the degree of $\mathcal{R}$.*

Since we have again considered the most general variant of the compressed word problems, we immediately obtain that also the word problem of any (fixed) automaton structure is in ExpSpace. On the other hand, we can combine the construction from the proof of Theorem 2.1.2.7 with some further ideas to obtain an automaton group with an ExpSpace-hard compressed word problem. Thus, there is no significantly better algorithm than the naïve approach given above.

ExpSpace-Hardness. The outline of the proof is the same as for the (normal) word problem: we start with a Turing machine M and construct a $\mathscr{G}$-automaton $\mathcal{T} = (Q, \Sigma, \delta)$ from it in the same way as in the proof of Theorem 2.1.2.7. This time, however, the Turing machine accepts an (arbitrary) ExpSpace-complete problem and we assume that the length of its configurations is $s(n) = n + 1 + 2^{n^e}$ (for some natural number e). Additionally, we assume the same normalizations as before.

Finally, we reduce the word problem of M to the compressed word problem of $\mathscr{G}(\mathcal{T})$. Here, we cannot simply use the same proof as in the case of the (normal) word problem, however! Recall that we have mapped the input word w of length n to the state sequence[67]

$$\boldsymbol{q} = B_0[f, \boldsymbol{q}_{s(n)}, \ldots, \boldsymbol{q}_1, \boldsymbol{c}', \boldsymbol{c}_{s(n)}, \ldots, \boldsymbol{c}_1, r]$$

[67] We continue to write B_0 for B_{β_0,α_0}.

for the reduction. The problem is that we now have exponentially many $\boldsymbol{c}_i$ and $\boldsymbol{q}_i$ and we, thus, cannot output all of them with a LogSpace (or even polynomial time) transducer – even if we compress every individual $\boldsymbol{c}_i$ and $\boldsymbol{q}_i$ using a straight-line program. On the positive side, we have that all $\boldsymbol{c}_i$ and all except linearly many $\boldsymbol{q}_i$ are structurally very similar: we have

$$\boldsymbol{c}_i = \overline{\checkmark_{\mathrm{id}}}^{i-1}\, \overline{\checkmark_{\mathrm{id}}} \checkmark_\sigma \checkmark_{\mathrm{id}}^{i-1} \text{ and } \boldsymbol{q}_j = \overline{\checkmark_{\mathrm{id}}}^{j-1}\, q_{[]} \checkmark_{\mathrm{id}}^{j-1}$$

for all $1 \leq i \leq s(n)$ and all $n+2 \leq j \leq s(n)$. Due to this structural similarity, we will still be able to output a single straight-line program that generates a word equal to $B_0[\boldsymbol{c}_{s(n)}, \dots, \boldsymbol{c}_{n+2}]$ in $\mathscr{G}(\mathcal{T})$ and one generating a word equal to $B_0[\boldsymbol{q}_{s(n)}, \dots, \boldsymbol{q}_{n+2}]$ in $\mathscr{G}(\mathcal{T})$.

Twisted Balanced Iterated Commutators. For the construction of these straight-line programs, we use a twisted version of our balanced iterated commutators, where the left side has an additional conjugation.

Definition 2.1.3.3. For $\alpha, \beta, \gamma \in \widetilde{Q}^*$ and $\boldsymbol{p} \in \widetilde{Q}^*$, we inductively define the word $B^\gamma_{\beta,\alpha}[\boldsymbol{p}, d]$ by

$$\begin{aligned} B^\gamma_{\beta,\alpha}[\boldsymbol{p}, 1] &= \boldsymbol{p} \quad \text{and} \\ B^\gamma_{\beta,\alpha}[\boldsymbol{p}, d] &= \left[\left(\bar{\gamma}^{\lfloor \frac{d}{2} \rfloor}\, B^\gamma_{\beta,\alpha}[\boldsymbol{p}, \lceil \tfrac{d}{2} \rceil]\ \gamma^{\lfloor \frac{d}{2} \rfloor} \right)^\beta,\ B^\gamma_{\beta,\alpha}[\boldsymbol{p}, \lfloor \tfrac{d}{2} \rfloor]^\alpha \right]. \end{aligned}$$

Compared to the (ordinary) balanced iterated commutators from Definition 2.1.2.3, the left side has an additional conjugation with $\gamma^{\lfloor \frac{d}{2} \rfloor}$.

Before we look further into how we can use the twisted balanced iterated commutators, we first have to return to the interaction between (ordinary) balanced iterated commutators and conjugation as we will need it in our proof. For normal commutators in groups, we have $[h, g]^k = [h^k, g^k]$ where g, h and k are group elements and, for balanced iterated commutators, we have something similar.

Fact 2.1.3.4. *Let $\alpha, \beta, \gamma \in \widetilde{Q}^*$ such that γ commutes with α and β in $\mathscr{G}(\mathcal{T})$. Then, we have*

$$B_{\beta,\alpha}[\boldsymbol{p}_d, \dots, \boldsymbol{p}_1]^\gamma = B_{\beta,\alpha}[\boldsymbol{p}_d^\gamma, \dots, \boldsymbol{p}_1^\gamma] \text{ in } \mathscr{G}(\mathcal{T})$$

for all $\boldsymbol{p}_d, \dots, \boldsymbol{p}_1 \in \widetilde{Q}^*$

Proof. We simply write B instead of $B_{\beta,\alpha}$ and prove the statement by induction. For $d = 1$, we have $B[\boldsymbol{p}_1]^\gamma = \boldsymbol{p}_1^\gamma = B[\boldsymbol{p}_1^\gamma]$ and, for $d > 1$, we have in $\mathscr{G}(\mathcal{T})$

$$\begin{aligned} B[\boldsymbol{p}_d, \dots, \boldsymbol{p}_1]^\gamma &= \left[B[\boldsymbol{p}_d, \dots, \boldsymbol{p}_{\lfloor \frac{d}{2} \rfloor + 1}]^\beta,\ B[\boldsymbol{p}_{\lfloor \frac{d}{2} \rfloor}, \dots, \boldsymbol{p}_1]^\alpha \right]^\gamma \\ &= \left[B[\boldsymbol{p}_d, \dots, \boldsymbol{p}_{\lfloor \frac{d}{2} \rfloor + 1}]^{\beta\gamma},\ B[\boldsymbol{p}_{\lfloor \frac{d}{2} \rfloor}, \dots, \boldsymbol{p}_1]^{\alpha\gamma} \right] && ([h,g]^k = [h^k, g^k]) \\ &= \left[B[\boldsymbol{p}_d, \dots, \boldsymbol{p}_{\lfloor \frac{d}{2} \rfloor + 1}]^{\gamma\beta},\ B[\boldsymbol{p}_{\lfloor \frac{d}{2} \rfloor}, \dots, \boldsymbol{p}_1]^{\gamma\alpha} \right] && (\gamma \text{ comutes with } \alpha, \beta) \\ &= \left[B[\boldsymbol{p}_d^\gamma, \dots, \boldsymbol{p}_{\lfloor \frac{d}{2} \rfloor + 1}^\gamma]^\beta,\ B[\boldsymbol{p}_{\lfloor \frac{d}{2} \rfloor}^\gamma, \dots, \boldsymbol{p}_1^\gamma]^\alpha \right] && (\text{by induction}) \\ &= B[\boldsymbol{p}_d^\gamma, \dots, \boldsymbol{p}_1^\gamma]. \end{aligned}$$

□

This compatibility between balanced iterated commutators and conjugation allows us to use the twisted version to move an iterated conjugation of the commutator entries into the commutator itself. As we have already seen, the check state sequences $\boldsymbol{c}_i$ and (most) $\boldsymbol{q}_i$ are of this form.

Lemma 2.1.3.5. *Let $\alpha, \beta, \gamma \in \widetilde{Q}^*$ such that γ commutes with α and β in $\mathscr{G}(\mathcal{T})$ and let $\boldsymbol{p} \in \widetilde{Q}^*$. Furthermore, let*

$$\boldsymbol{p}_d = \overline{\gamma}^{d-1} \boldsymbol{p} \gamma^{d-1}$$

for $d \geq 1$. Then, we have

$$B^{\gamma}_{\beta,\alpha}[\boldsymbol{p}, d] = B_{\beta,\alpha}[\boldsymbol{p}_d, \ldots, \boldsymbol{p}_1] \text{ in } \mathscr{G}(\mathcal{T}).$$

Proof. Similar to before, we write B^{γ} for $B^{\gamma}_{\beta,\alpha}$ and B for $B_{\beta,\alpha}$ and prove the statement by induction on d. For $d = 1$, we have $B^{\gamma}[\boldsymbol{p}, 1] = \boldsymbol{p} = \boldsymbol{p}_1 = B[\boldsymbol{p}_1]$ and, for $d > 1$, we have

$$\begin{aligned}
B^{\gamma}[\boldsymbol{p}, d] &= \left[\left(\overline{\gamma}^{\lfloor \frac{d}{2} \rfloor} B^{\gamma}[\boldsymbol{p}, \lceil \tfrac{d}{2} \rceil] \ \gamma^{\lfloor \frac{d}{2} \rfloor} \right)^{\beta}, \ B^{\gamma}[\boldsymbol{p}, \lfloor \tfrac{d}{2} \rfloor]^{\alpha} \right] \\
&= \left[\left(\overline{\gamma}^{\lfloor \frac{d}{2} \rfloor} B[\boldsymbol{p}_{\lceil \frac{d}{2} \rceil}, \ldots, \boldsymbol{p}_1] \gamma^{\lfloor \frac{d}{2} \rfloor} \right)^{\beta}, \ B[\boldsymbol{p}_{\lfloor \frac{d}{2} \rfloor}, \ldots, \boldsymbol{p}_1]^{\alpha} \right] && \text{(induction)} \\
&= \left[\left(B[\overline{\gamma}^{\lfloor \frac{d}{2} \rfloor} \boldsymbol{p}_{\lceil \frac{d}{2} \rceil} \gamma^{\lfloor \frac{d}{2} \rfloor}, \ldots, \overline{\gamma}^{\lfloor \frac{d}{2} \rfloor} \boldsymbol{p}_1 \gamma^{\lfloor \frac{d}{2} \rfloor}] \right)^{\beta}, \ B[\boldsymbol{p}_{\lfloor \frac{d}{2} \rfloor}, \ldots, \boldsymbol{p}_1]^{\alpha} \right] && \text{(Fact 2.1.3.4)} \\
&= \left[\left(B[\boldsymbol{p}_d, \ldots, \boldsymbol{p}_{1+\lfloor \frac{d}{2} \rfloor}] \right)^{\beta}, \ B[\boldsymbol{p}_{\lfloor \frac{d}{2} \rfloor}, \ldots, \boldsymbol{p}_1]^{\alpha} \right] && \text{(definition of } \boldsymbol{p}_i) \\
&= B[\boldsymbol{p}_d, \ldots, \boldsymbol{p}_1] && \text{(definition of } B)
\end{aligned}$$

in $\mathscr{G}(\mathcal{T})$. □

The connection in Lemma 2.1.3.5 allows us to use twisted balanced iterated commutators for the check sequences $\boldsymbol{c}_i$ and $\boldsymbol{q}_i$. The advantage of this approach is that the twisted version can efficiently be compressed into straight-line programs (although the corresponding (ordinary) balanced iterated commutator would have too many entries).

Lemma 2.1.3.6. *On input $\alpha, \beta, \gamma \in \widetilde{Q}^*$, $\boldsymbol{p} \in \widetilde{Q}^*$ and $\ell \in \mathbb{N}$ in unary, one can compute in logarithmic space a straight-line program $\mathcal{G}$ with $\mathscr{W}(\mathcal{G}) = B^{\gamma}_{\beta,\alpha}[\boldsymbol{p}, 2^{\ell}]$.*

Proof. The alphabet of the straight-line program is obviously $\widetilde{Q}$ and we only give the variables implicitly. Clearly, if we can compute the production rules for a variable X generating $\mathscr{W}(X)$, then we can also compute the production rules for a variable $\overline{X}$ with $\mathscr{W}(\overline{X}) = \overline{\mathscr{W}(X)}$. Therefore, we only give the positive version for every variable (but always assume that we also have a negative one).

First, we add the production rules for

$$M_{2^{\ell-1}} \to M_{2^{\ell-2}} M_{2^{\ell-2}}, \quad \ldots, \quad M_{2^1} \to M_{2^0} M_{2^0}, \quad M_{2^0} \to \gamma$$

because we need blocks of γ for the recursion of $B^{\gamma}_{\beta,\alpha}$. Clearly, we have $\mathscr{W}(M_{2^d}) = \gamma^{2^d}$ for all $0 \leq d < \ell$. For the actual commutator, we use the variables A_{2^d} for $0 \leq d \leq \ell$ and add the production rules for $A_{2^0} \to \boldsymbol{p}$ and

$$\begin{aligned} A_{2^d} \to\ & \overline{\beta}\ \overline{M}_{2^{d-1}} \overline{A}_{2^{d-1}} M_{2^{d-1}}\ \beta\ \overline{\alpha} \overline{A}_{2^{d-1}} \alpha \\ & \overline{\beta}\ \overline{M}_{2^{d-1}} A_{2^{d-1}} M_{2^{d-1}}\ \beta\ \overline{\alpha} A_{2^{d-1}} \alpha \end{aligned}$$

for all $0 < d \leq \ell$. Using a simple induction it is now easy to see that we have $\mathscr{W}(A_{2^d}) = B^{\gamma}_{\beta,\alpha}[\boldsymbol{p}, 2^d]$. Accordingly, we choose A_{2^ℓ} as the start variable.

To compute the productions rules, we obviously only need to count up to ℓ and this can clearly be done in logarithmic space as ℓ is given in unary. □

With twisted balanced iterated commutators and the corresponding straight-line programs, we have introduced the missing piece to adapt the proof for PSpace and the (normal) word problem from Theorem 2.1.2.7 to ExpSpace and the compressed word problem.

Theorem 2.1.3.7 ([5, Theorem 13])**.** *There is an automaton group whose compressed word problem*

Constant:	*a $\mathscr{G}$-automaton $\mathcal{T} = (Q, \Sigma, \delta)$*
Input:	*a straight-line program $\mathcal{P}$ over $\widetilde{Q}$*
Question:	*is $\mathscr{W}(\mathcal{P}) = \mathbb{1}$ in $\mathscr{G}(\mathcal{T})$?*

is ExpSpace*-complete.*

Proof. We only have to show that the problem is ExpSpace-hard by Corollary 2.1.3.2. As already mentioned, we construct the $\mathscr{G}$-automaton $\mathcal{T} = (Q, \Sigma, \delta)$ from the machine M for an ExpSpace-complete problem in the same way as in the proof of Theorem 2.1.2.7 and reduce the (normal) word problem of M to the compressed word problem of $\mathscr{G}(\mathcal{T})$. We map an input word w of length n for M to a straight-line program for a state sequence equal to

$$B_0\Big[f, B_0[\boldsymbol{q}_{s(n)}, \ldots, \boldsymbol{q}_{n+2}], \boldsymbol{q}_{n+1}, \ldots, \boldsymbol{q}_1, \boldsymbol{c}', B_0[\boldsymbol{c}_{s(n)}, \ldots, \boldsymbol{c}_{n+2}], \boldsymbol{c}_{n+1}, \ldots, \boldsymbol{c}_1, r\Big]$$

in $\mathscr{G}(\mathcal{T})$. If we think of the balanced iterated commutator as a logical conjunction, we see that the inner balanced iterated commutators do not change the semantics compared to the state sequence used as input for the (normal) word problem in the proof of Theorem 2.1.2.7.

Thus, it remains to describe how the straight-line program can be computed in logarithmic space. If we have straight-line programs for the individual entries, we immediately also obtain straight-line programs for their inverses and can combine everything into a straight-line program for the overall balanced iterated commutator. This can be done (on the level of the variables) in logarithmic space by Lemma 2.1.2.6. For f, $\boldsymbol{q}_{n+1}, \ldots, \boldsymbol{q}_1$, $\boldsymbol{c}_{n+1}, \ldots, \boldsymbol{c}_1$ and r, we do not even need straight-line programs but can output the words directly (as in the PSpace-case).

For the inner balanced iterated commutators, we recall that we have

$$\boldsymbol{q}_{n+1+d} = \overline{\checkmark_{id}}^{d-1}\ \overline{\checkmark_{\mathrm{id}}}^{n+1} q_{\square} \checkmark_{id}^{n+1}\ \checkmark_{id}^{d-1} \quad \text{and} \quad \boldsymbol{c}_{n+1+d} = \overline{\checkmark_{id}}^{d-1}\ \overline{\checkmark_{\mathrm{id}}}^{n+1} \overline{\checkmark_{\mathrm{id}}} \checkmark_{\sigma} \checkmark_{\mathrm{id}}^{n+1}\ \checkmark_{\mathrm{id}}^{d-1}$$

for all $1 \leq d \leq 2^{n^e}$. Thus, we have

$$B_0^{\checkmark_{\mathrm{id}}}[\overline{\checkmark_{\mathrm{id}}}^{n+1} q_{\square} \checkmark_{\mathrm{id}}^{n+1}, 2^{n^e}] = B_0[\boldsymbol{q}_{\underbrace{2^{n^e}+n+1}_{=s(n)}}, \ldots, \boldsymbol{q}_{n+2}] \quad \text{in } \mathscr{G}(\mathcal{T})$$

and

$$B_0^{\checkmark_{\mathrm{id}}}[\overline{\checkmark_{\mathrm{id}}}^{n+1}\ \overline{\checkmark_{\mathrm{id}}} \checkmark_{\sigma}\ \checkmark_{\mathrm{id}}^{n+1}, 2^{n^e}] = B_0[\boldsymbol{c}_{\underbrace{2^{n^e}+n+1}_{=s(n)}}, \ldots, \boldsymbol{c}_{n+2}] \quad \text{in } \mathscr{G}(\mathcal{T})$$

by Lemma 2.1.3.5 where we write $B_0^{\checkmark_{\mathrm{id}}}$ for $B_{\beta_0,\alpha_0}^{\checkmark_{\mathrm{id}}}$. In order to apply Lemma 2.1.3.5, we need that $\checkmark_{\mathrm{id}}$ commutes with α_0 and β_0. However, this immediately follows from the construction of $\mathcal{T}$ (as $\checkmark_{\mathrm{id}}$ only manipulates the TM-part of the input word while α_0 and β_0 only manipulate the A_5-part). Finally, we can compute a straight-line program for the two twisted balanced iterated commutators in LogSpace by Lemma 2.1.3.6. Here, it is important that n is given in unary and that, thus, n^e can also be output in unary.

The last remaining part is a straight-line program for

$$\boldsymbol{c}' = \overline{\checkmark_{\mathrm{id}}}^{s(n)} c \checkmark_{\mathrm{id}}^{s(n)} = \overline{\checkmark_{\mathrm{id}}}^{2^{n^e}}\ \overline{\checkmark_{\mathrm{id}}}^{n+1} c \checkmark_{\mathrm{id}}^{n+1}\ \checkmark_{\mathrm{id}}^{2^{n^e}}.$$

The inner part can be output directly and the outer $\checkmark_{\mathrm{id}}$-blocks of length 2^{n^e} can be generated in the same way as in the proof of Lemma 2.1.3.6.[68] □

We can take the direct product of the automaton group with a PSpace-complete word problem and the automaton group with an ExpSpace-complete compressed word problem.[69] In this way, we obtain an automaton group whose (normal) word problem is PSpace-complete and whose compressed word problem is ExpSpace-complete and, thus, provably harder (by the space hierarchy theorem, see e. g. [Pap94, Theorem 7.2, p. 145]).

Corollary 2.1.3.8 ([5, Corollary 14])**.** *There is an automaton group whose word problem*

Constant:	*a $\mathscr{G}$-automaton $\mathcal{T} = (Q, \Sigma, \delta)$*
Input:	*a state sequence $\boldsymbol{q} \in \widetilde{Q}^*$*
Question:	*is $\boldsymbol{q} = \mathbb{1}$ in $\mathscr{G}(\mathcal{T})$?*

is PSpace*-complete and whose compressed word problem*

Constant:	*a $\mathscr{G}$-automaton $\mathcal{T} = (Q, \Sigma, \delta)$*
Input:	*a straight-line program $\mathcal{P}$ over $\widetilde{Q}$*
Question:	*is $\mathscr{W}(\mathcal{P}) = \mathbb{1}$ in $\mathscr{G}(\mathcal{T})$?*

is ExpSpace*-complete.*

[68] In fact, we already have the required production rules and variables up to $M_{2^{n^e-1}}$.

[69] An alternative way to obtain this result is to use a universal Turing machine for M which allows to simulate a polynomially space-bounded machine in polynomial space and an exponentially space-bounded one in exponential space (i. e. the simulation must not introduce too much space overhead). Then, we can simply construct the $\mathscr{G}$-automaton $\mathcal{T}$ from this universal machine.

2.2 Positive Relations

In this section, we are going to study the problem of finding a so-called positive relation in an automaton group. For an $\mathscr{S}$-automaton $\mathcal{T} = (Q, \Sigma, \delta)$, we define the set of *positive relations* as

$$\mathscr{P}(\mathcal{T}) = \{\boldsymbol{q} \in Q^+ \mid \boldsymbol{q} =_{\mathcal{T}} \varepsilon\}.$$

The name stems from the fact that, if $\mathcal{T}$ is a $\mathscr{G}$-automaton, then $\mathscr{P}(\mathcal{T})$ is the set of words in the **positive** generators of the group generated by $\mathcal{T}$ which are equal to the neutral element. While the set of positive relations is obviously connected to the freeness of the generated group, it is not an algebraic property but depends on the presentation. However, we will see that it also has some (somewhat surprising) connections to algebraic properties of the semigroup generated by $\mathcal{T}$.

We will see that we cannot algorithmically check the existence of positive relations for a given $\mathscr{G}$-automaton. In other words, we show that the problem Automaton Group Positive Relation

Input: a $\mathscr{G}$-automaton $\mathcal{T}$
Question: is $\mathscr{P}(\mathcal{T}) \neq \emptyset$?

is undecidable. Afterwards, we will look into some other decision problems that turn out to be equivalent to Automaton Group Positive Relation. The first of these problems is to check whether the semigroup generated by a given $\mathscr{S}$-automaton (or, in fact, $\mathscr{G}$-automaton) is a monoid. We obtain that the problem Automaton Semigroup Neutrality

Input: an $\mathscr{S}$-automaton $\mathcal{T}$
Question: does $\mathscr{S}(\mathcal{T})$ have a neutral element?

is undecidable. Then, we will see that the existence of a positive relation is closely connected to the existence of a torsion element in the semigroup generated by an $\mathscr{S}$-automaton. This will lead to the undecidability of the problem Torsion Element

Input: an $\mathscr{S}$-automaton $\mathcal{T}$
Question: does $\mathscr{S}(\mathcal{T})$ contain an element of torsion?

Finally, we use the connection between elements of torsion and finite orbits in the dual already given in Theorem 1.4.2.3 to obtain that the problem Finite Orbit

Input: an $\mathscr{S}$-automaton $\mathcal{T} = (Q, \Sigma, \delta)$
Question: $\exists \alpha \in \Sigma^\omega : |Q^* \circ \alpha| < \infty$?

is undecidable. Because the finiteness problem for automaton semigroups is co-equivalent to finding an ω-word with an infinite orbit by Corollary 1.4.1.14, Finite Orbit can be seen as a dual problem to it.

The interest in finding positive relations in an automaton group is obviously also motivated by the open problem to decide whether a given automaton semigroup or group is free [GNS00, 7.2 b)]. However, the approach we will be taking here to show the undecidability of Automaton Group Positive Relation does not seem to be well suited to solve these problems unfortunately. We will discuss this briefly at the end of this section.

Attribution. The results concerning Automaton Group Positive Relation are joint work with Daniele D'Angeli and Emanuele Rodaro [3]. Crucial for the reductions is an automaton construction used by Šunić and Ventura to construct an automaton group with undecidable conjugacy problem [ŠV12], which the next subsection is devoted to. This construction uses techniques similar to those used by Brunner and Sidki [BS98] to present the generators of the affine group over $\mathbb{Z}^d$ as finite state automorphisms. Finally, we use a variant of Post's Correspondence Problem, called *Identity Correspondence Problem*, for the reduction. This has been shown to be undecidable by Bell and Potapov [BP10].

The relations between Automaton Group Positive Relation and the other stated problems have not been discussed in this form previously but the results on the non-applicability of the presented approach for the freeness problems of automaton groups and semigroups are again joint work with Daniele D'Angeli and Emanuele Rodaro [6] and the presentation loosely follows the one given there.

2.2.1 A Construction Due to Šunić and Ventura

We will investigate a construction by Šunić and Ventura to present d-dimensional linear – or, in fact, affine – transformations over the ring of integers using automata. While it was used by Šunić and Ventura to construct an automaton group with undecidable conjugacy problem [ŠV12], the construction is more versatile. The constructed automata act on n-adic expansions of vectors over the ring of n-adic integers. This is where we start our study.

The Ring of n-Adic Integers. Let $n \geq 2$ be a natural number. The ring of *n-adic integers* is the projective limit

$$\mathbb{Z}_n = \varprojlim_{k \geq 1} \mathbb{Z}/n^k\mathbb{Z}.$$

Thus, its elements are sequences $(a_k + n^k\mathbb{Z})_{k\geq 1}$ of residue classes $a_k + n^k\mathbb{Z} \in \mathbb{Z}/n^k\mathbb{Z}$ satisfying $a_k \equiv a_\ell \mod n^k$ for all $\ell \geq k$ and its operations are the component-wise addition and multiplication of these sequences. The ring of (normal) integers $\mathbb{Z}$ embeds into the ring of n-adic integers $\mathbb{Z}_n$ by mapping $z \in \mathbb{Z}$ to the sequence $(z + n\mathbb{Z}, z + n^2\mathbb{Z}, \dots)$. If z is a natural number, then the corresponding sequence of the least non-negative representatives for these residue classes will eventually become stationary and the stationary value is z. On the other hand, any such stationary sequence belongs to a natural number.

An alternative way to represent n-adic integers is to use formal sums of the form

$$Z = \sum_{k=0}^{\infty} d_k n^k$$

with $0 \leq d_k < n$. It is natural to write $Z \equiv \sum_{k=0}^{\ell-1} d_k n^k \mod n^\ell$ and, so, the sum Z is considered to represent the n-adic integer

$$(d_0,\ d_0 + d_1 n,\ d_0 + d_1 n + d_2 n^2,\ \dots).$$

On the other hand, any n-adic integer $(a_1 + n\mathbb{Z}, a_2 + n^2\mathbb{Z}, \dots)$ can be written as the sum

$$\sum_{k=0}^{\infty} d_k n^k \quad \text{with } d_k = \frac{a_{k+1} - a_k}{n^k}$$

where we set $a_0 = 0$ and assume without loss of generality that all other a_k are the least non-negative representatives of their classes. Together with $a_k \equiv a_{k+1} \mod n^k$, this yields $0 \leq a_k \leq a_{k+1}$ and that d_k is a natural number. Since we also have $a_{k+1} < n^{k+1}$, we obtain $0 \leq d_k < n$.

Instead of writing the sum $Z = \sum_{k=0}^{\infty} d_k n^k$, we rather use its *$n$-adic expansion*: this is the ω-word $d_0 d_1 \dots$ over the alphabet $\{0, \dots, n-1\}$.

Example 2.2.1.1. We have the following presentations of integers as 2-adic numbers.

number	sequence	2-adic expansion
-2	$(-2 + 2\mathbb{Z}, -2 + 4\mathbb{Z}, -2 + 8\mathbb{Z}, \dots)$	$0111\dots$
	$= (0 + 2\mathbb{Z}, 2 + 4\mathbb{Z}, 6 + 8\mathbb{Z}, \dots)$	
-1	$(-1 + 2\mathbb{Z}, -1 + 4\mathbb{Z}, -1 + 8\mathbb{Z}, \dots)$	$1111\dots$
	$= (1 + 2\mathbb{Z}, 3 + 4\mathbb{Z}, 7 + 8\mathbb{Z}, \dots)$	
0	$(0 + 2\mathbb{Z}, 0 + 4\mathbb{Z}, 0 + 8\mathbb{Z}, \dots)$	$0000\dots$
1	$(1 + 2\mathbb{Z}, 1 + 4\mathbb{Z}, 1 + 8\mathbb{Z}, \dots)$	$1000\dots$
2	$(2 + 2\mathbb{Z}, 2 + 4\mathbb{Z}, 2 + 8\mathbb{Z}, \dots)$	$0100\dots$
	$= (0 + 2\mathbb{Z}, 2 + 4\mathbb{Z}, 2 + 8\mathbb{Z}, \dots)$	
3	$(3 + 2\mathbb{Z}, 3 + 4\mathbb{Z}, 3 + 8\mathbb{Z}, \dots)$	$1100\dots$
	$= (1 + 2\mathbb{Z}, 3 + 4\mathbb{Z}, 3 + 8\mathbb{Z}, \dots)$	

From this table, it is easy to observe that the adding machine from Example 0.2.1.4 is actually operating on 2-adic expansions of integers!

We will shortly be working in the module $\mathbb{Z}_n^d$ with dimension d rather than in the ring $\mathbb{Z}_n$ directly. In a slight abuse of nomenclature, we will call the elements of $\mathbb{Z}_n^d$ *vectors*. Let

$$\boldsymbol{x} = \begin{pmatrix} x_1 \\ \vdots \\ x_d \end{pmatrix} \in \mathbb{Z}_n^d$$

be such a vector and let $x_{i,1}x_{i,2}x_{i,3}\dots$ be the n-adic expansion of its component x_i. We could present the vector as a vector of these n-adic expansions:

$$\begin{pmatrix} x_{1,1}x_{1,2}x_{1,3}\dots \\ \vdots \\ x_{d,1}x_{d,2}x_{d,3}\dots \end{pmatrix}$$

However, for the automaton presentation later, it will be more convenient to define the *n-adic expansion of the vector* $\boldsymbol{x}$ as the ω-word

$$\begin{pmatrix} x_{1,1} \\ \vdots \\ x_{d,1} \end{pmatrix} \begin{pmatrix} x_{1,2} \\ \vdots \\ x_{d,2} \end{pmatrix} \begin{pmatrix} x_{1,3} \\ \vdots \\ x_{d,3} \end{pmatrix} \dots$$

whose letters are from $\{0, \dots, n-1\}^d$.

Affine Integer Transformations of n-Adic expansions. When using the sequence presentation of n-adic integers, we can add and multiply component-wise. Performing the same operations using the n-adic expansion is a bit less obvious. We will describe the operation in the more general setting of applying affine transformations.

Let $M \in \mathbb{Z}^{d\times d}$ be a matrix and $\boldsymbol{c} \in \mathbb{Z}^d$ a vector with (normal) integer coefficients as entries. They induce an affine transformation

$$\begin{aligned} M_{\boldsymbol{c}} : \mathbb{Z}_n^d &\to \mathbb{Z}_n^d \\ \boldsymbol{x} &\mapsto \boldsymbol{c} + M\boldsymbol{x}. \end{aligned}$$

For example using the sequence presentation of n-adic integers, it is easy to see that $M_{\boldsymbol{c}}$ is equal to another affine transformation $M'_{\boldsymbol{c}'}$ with $M' \in \mathbb{Z}^{d\times d}$ and $\boldsymbol{c}' \in \mathbb{Z}^d$ if and only if they coincide on all vectors from $\mathbb{Z}^d$, i. e. on those with (normal) integer coefficients.

In order to see how this affine transformation operates on the n-adic expansion $x_1x_2x_3\dots$ with $x_1, x_2, \dots \in \{0, \dots n-1\}^d$ of some vector $\boldsymbol{x} \in \mathbb{Z}_n^d$, we first define the operators mod and div. For an integer $z \in \mathbb{Z}$, we define $z \bmod n$ as the least non-negative representative of the residue class $z + n\mathbb{Z}$ and $z \operatorname{div} n$ as $\lfloor \frac{z}{n} \rfloor$. When applied to a vector, mod and div operate component-wise.

Recall that the n-adic expansion $x_1x_2x_3\dots$ of $\boldsymbol{x}$ is a different presentation for the sum $\sum_{k=0}^{\infty} x_k n^k$. With this in mind, we have (see [ŠV12, Lemma 4.3])

$$\begin{aligned} M_{\boldsymbol{c}}(x_1x_2x_3\dots) &= \boldsymbol{c} + M(x_1x_2x_3\dots) \\ &= \boldsymbol{c} + M\big(x_1 + n(x_2x_3\dots)\big) \\ &= \boldsymbol{c} + Mx_1 + n\big(M(x_2x_3\dots)\big) \\ &= M_{\boldsymbol{c}}x_1 + n\big(M(x_2x_3\dots)\big) \\ &= [M_{\boldsymbol{c}}x_1 \bmod n] + n[M_{\boldsymbol{c}}x_1 \operatorname{div} n] + n\big(M(x_2x_3\dots)\big) \\ &= [M_{\boldsymbol{c}}x_1 \bmod n] + n[(\boldsymbol{c} + Mx_1) \operatorname{div} n] + n\big(M(x_2x_3\dots)\big) \\ &= [M_{\boldsymbol{c}}x_1 \bmod n] + n[M_{(\boldsymbol{c}+Mx_1) \operatorname{div} n}(x_2x_3\dots)], \end{aligned}$$

which means that the first letter of the n-adic expansion gets transformed into $M_{\boldsymbol{c}}x_1 \bmod n \in \{0, \dots, n-1\}$ and, for the subsequent letters, we have to apply an affine transformation with a new "carry vector" $(\boldsymbol{c} + Mx_1) \operatorname{div} n \in \mathbb{Z}^d$. We can reproduce this behavior with a finite state automaton if we can bound this carry vector. Let $m_{i,j} \in \mathbb{Z}$ denote the

entry in the i^{th} row and j^{th} column of M and define $\|M\| = \max_{1\le i\le d} \sum_{j=1}^{d} |m_{i,j}|$ as the maximum absolute row sum norm of M. We define the finite set

$$C_M = \left\{ \boldsymbol{c} = \begin{pmatrix} c_1 \\ \vdots \\ c_d \end{pmatrix} \in \mathbb{Z}^d \;\middle|\; -\|M\| \le c_1, \ldots, c_d \le \|M\| - 1 \right\}$$

and claim that, if we start with a vector $\boldsymbol{c}$ from C_M, then the "new" carry vector $\boldsymbol{c}' = \boldsymbol{d}' \operatorname{div} n \in \mathbb{Z}^d$ with $\boldsymbol{d}' = (\boldsymbol{c} + Mx_1)$ is in C_M as well. Let c_i, c_i', d_i' and y_i be the i^{th} component of $\boldsymbol{c}$, $\boldsymbol{c}'$, $\boldsymbol{d}'$ and x_1, respectively. We have

$$-n\|M\| = -\|M\| - \|M\|(n-1) \le d_i' = c_i + \sum_{j=1}^{d} m_{i,j} y_j \le \|M\| - 1 + \|M\|(n-1) = n\|M\| - 1.$$

This implies $-\|M\| \le c_i' \operatorname{div} n \le \|M\| - 1$ and, thus, $\boldsymbol{c}' \in C_M$, which proves our claim. Finally, we observe that C_M always contains the zero vector $\mathbf{0} \in \mathbb{Z}^d$.

The Automaton Construction. The above considerations allow us to define the *Šunić-Ventura automaton* $\mathcal{T}_{M,n}$ for M with respect to n: its state set is

$$Q_M = \{ s_{M,\boldsymbol{c}} \mid \boldsymbol{c} \in C_M \},$$

its alphabet is $\{0, \ldots, n-1\}^d$ and it contains the transitions

$$s_{M,\boldsymbol{c}} \xrightarrow{x/(\boldsymbol{c}+Mx) \bmod n} s_{M,(\boldsymbol{c}+Mx) \operatorname{div} n}$$

for all $x \in \{0, \ldots, n-1\}^d$ and $\boldsymbol{c} \in C_M$. Furthermore, we define the Šunić-Ventura automaton with respect to n of a finite set $\mathcal{M}$ of matrices from $\mathbb{Z}^{d\times d}$ as the union automaton

$$\mathcal{T}_{\mathcal{M},n} = \biguplus_{M\in\mathcal{M}} \mathcal{T}_{M,n}.$$

These automata are clearly deterministic and complete, and the action of $s_{M,\boldsymbol{c}}$ on the n-adic expansion of a vector $\boldsymbol{x} \in \mathbb{Z}_n^d$ is the same as applying $M_{\boldsymbol{c}}$ by construction. In particular, the action of $s_{M,\mathbf{0}}$ is the same as applying M. Accordingly, the semigroup generated by $\mathcal{T}_{M,n}$ is (isomorphic to) the closure of $\{M_{\boldsymbol{c}} \mid \boldsymbol{c} \in C_M\}$ under composition.

If $M \in \mathbb{Z}^{d\times d}$ is invertible,[70] we can choose n co-prime to $\det M$. In this case, M is also invertible as a matrix from $(\mathbb{Z}/n\mathbb{Z})^{d\times d}$ and the Šunić-Ventura automaton for M is invertible. Again, by construction, the action of the state $\overline{s_{M,\mathbf{0}}}$ in the inverse automaton $\overline{\mathcal{T}}_{M,n}$ on an n-adic expansion of a vector is the same as applying M^{-1} to this vector.

[70] If we say that a matrix M from $\mathbb{Z}^{d\times d}$ or $\mathbb{N}^{d\times d}$ is invertible, we mean that it has non-zero determinant, i. e. its inverse is not necessarily from $\mathbb{Z}^{d\times d}$ or $\mathbb{N}^{d\times d}$, respectively.

The Affine Semigroup and the Affine Group. The semigroup $\mathrm{SGL}_d(\mathbb{Z}_n)$ of matrices from $\mathbb{Z}_n^{d\times d}$ acts naturally (from the left) on the additive group $\mathbb{Z}_n^d$ via matrix multiplication. This allows us to define the *affine semigroup* $\mathrm{SAff}_d(\mathbb{Z}_n) = \mathbb{Z}_n^d \rtimes \mathrm{SGL}_d(\mathbb{Z}_n)$ and to identify the affine transformations $M_{\boldsymbol{c}}$ used above with the elements $(\boldsymbol{c}, M)$ from this semigroup. Under this view, the semigroup generated by the Šunić-Ventura automaton $\mathcal{T}_{\mathcal{M},n}$ is

(isomorphic to) a subsemigroup of $\mathrm{SAff}_d(\mathbb{Z}_n)$. In fact, as long as we do not consider inverses, the generated semigroup is even a subsemigroup of $\mathrm{SAff}_d(\mathbb{Z}) = \mathbb{Z}^d \rtimes \mathrm{SGL}_d(\mathbb{Z})$ (which is defined analogously but all components are from $\mathbb{Z}$ instead of $\mathbb{Z}_n$).

To define the affine group, we first need to define the semidirect product of groups. Let a group K act (from the left) on another group H, i. e. there is a group homomorphism $\alpha : K \to \operatorname{Aut} H, k \mapsto \alpha_k$ from K to the automorphism group $\operatorname{Aut} H$ of H. Then we can define the semidirect product $H \rtimes_\alpha K$ of the two groups. Its elements are from $H \times K$ and its multiplication is given by $(h_1, k_1)(h_2, k_2) = (h_1 \alpha_{k_1}(h_2), k_1 k_2)$. As with semigroups, we simply write $H \rtimes K$ instead of $H \rtimes_\alpha K$ if the action is implicitly given by the context.

Now, the group $\mathrm{GL}_d(\mathbb{Z}_n)$ of matrices from $\mathbb{Z}_n^{d\times d}$ with determinant co-prime to n acts on the additive group $\mathbb{Z}_n^d$ as a group (again, via matrix multiplication), which allows us to define the *affine group* $\mathrm{Aff}_d(\mathbb{Z}_n) = \mathbb{Z}_n^d \rtimes \mathrm{GL}_d(\mathbb{Z}_n)$. By construction, if $\mathcal{M}$ contains (finitely many) matrices with determinant co-prime to n, then the group generated by $\mathcal{T}_{\mathcal{M},n}$ is (isomorphic to) a subgroup of $\mathrm{Aff}_d(\mathbb{Z}_n)$. In fact, it is even a subgroup of $\mathbb{Z}_n^d \rtimes \mathscr{G}(\mathcal{M})$ where $\mathscr{G}(\mathcal{M})$ is the group generated by the matrices from $\mathcal{M}$. In this case, it contains $\mathscr{G}(\mathcal{M})$ as a subgroup (because we have the zero vector $\mathbf{0}$ in all C_M). If the matrices have all either determinant -1 or 1 (i. e. if they are elements from $\mathrm{GL}_d(\mathbb{Z})$), then the group generated by $\mathcal{T}_{\mathcal{M},n}$ is isomorphic to $\mathbb{Z}^d \rtimes \mathscr{G}(\mathcal{M})$ [ŠV12, Lemma 4.1].

2.2.2 The Reduction

With the automaton construction by Šunić and Ventura at hand, we can prove the undecidability of the above mentioned problems. For Automaton Group Positive Relation, we will do this in two steps. First, we reduce the Identity Correspondence Problem ICP[71]

[71] Recall that $\mathscr{F}(\Sigma)$ denotes the free group over Σ.

Constant: a binary alphabet $\Sigma = \{a, b\}$
Input: a finite number of pairs $(g_1, h_1), \ldots, (g_m, h_m) \in \mathscr{F}(\Sigma) \times \mathscr{F}(\Sigma)$
Question: is there a finite, non-empty sequence $i_1, i_2, \ldots, i_\ell$ of indices $1 \leq i_1, i_2, \ldots, i_\ell \leq m$ with

$$g_{i_1} g_{i_2} \ldots g_{i_\ell} = h_{i_1} h_{i_2} \ldots h_{i_\ell} = \mathbb{1}_{\mathscr{F}(\Sigma)}?$$

to a variant of Automaton Group Positive Relation for matrix groups, called Matrix Group Positive Relation. Then, we will reduce Matrix Group Positive Relation to Automaton Group Positive Relation. Because ICP was shown to be undecidable by Bell and Potapov [BP10, Theorem 11], this shows that Automaton Group Positive Relation is also undecidable.

For a finite set $\mathcal{M}$ of matrices with the same dimension d, let

$$\mathscr{P}(\mathcal{M}) = \{M_1 M_2 \ldots M_\ell \in \mathcal{M}^+ \mid M_1 M_2 \ldots M_\ell = 1\}$$

where 1 denotes the $d \times d$ identity matrix and define Matrix Group Positive Relation as the problem

Constant: a natural number d
Input: a finite set $\mathcal{M}$ of invertible matrices from $\mathbb{Z}^{d\times d}$
Question: is $\mathscr{P}(\mathcal{M}) \neq \emptyset$?

Proposition 2.2.2.1. MATRIX GROUP POSITIVE RELATIONS *is undecidable for* $d \geq 4$.

Proof (see also proof of [3, Theorem 3.7]). As already mentioned, we give a reduction from ICP to MATRIX GROUP POSITIVE RELATION. For this, we use the usual embedding ϱ of the free group $\mathscr{F}(\Sigma)$ into the special linear group $SL(2,\mathbb{Z})$ given by

$$\varrho(a) = \begin{pmatrix} 1 & 2 \\ 0 & 1 \end{pmatrix} \quad \text{and} \quad \varrho(b) = \begin{pmatrix} 1 & 0 \\ 2 & 1 \end{pmatrix}$$

and encode each pair (g_i, h_i) as the 4×4 block matrix

$$M_i = \begin{pmatrix} \varrho(g_i) & O_2 \\ O_2 & \varrho(h_i) \end{pmatrix}$$

where O_2 denotes the 2×2 zero matrix. As the input for MATRIX GROUP POSITIVE RELATION, we use $\mathcal{M} = \{M_1, \ldots, M_m\}$, which is clearly computable and contains invertible matrices from $\mathbb{Z}^{4\times 4}$. By construction, we also clearly have that every solution to the input ICP instance induces a positive relation in $\mathscr{P}(\mathcal{M})$ and vice-versa. □

For finally reducing MATRIX GROUP POSITIVE RELATION to AUTOMATON GROUP POSITIVE RELATION, we use Šunić and Ventura's construction from above and the following connection.

Lemma 2.2.2.2 (see [3, Lemma 3.6])**.** *Let $\mathcal{M}$ be a finite set of matrices from $\mathbb{Z}^{d\times d}$ for some d, $n \geq 2$ and let $M_k \ldots M_1$ and $N_\ell \ldots N_1$ be two sequences of matrices from $\mathcal{M}$. Then, we have*

$$M_k \ldots M_1 = N_\ell \ldots N_1$$

if and only if there are two sequences $p_1 \ldots p_k$ and $q_1 \ldots q_\ell$ in which p_i is a state of the Šunić-Ventura automaton $\mathcal{T}_{M_i,n}$ for all $1 \leq i \leq k$ and q_j is a state of $\mathcal{T}_{N_j,n}$ for all $1 \leq j \leq \ell$ with

$$p_k \ldots p_1 =_{\mathcal{T}_{\mathcal{M},n}} q_\ell \ldots q_1.$$

In particular, we have $M_k \ldots M_1 = 1$ if and only if there is a sequence $p_k \ldots p_1$ in which p_i is a state of $\mathcal{T}_{M_i,k}$ for all $1 \leq i \leq n$ with

$$p_k \ldots p_1 =_{\mathcal{T}_{\mathcal{M},n}} \varepsilon,$$

i. e. $p_k \ldots p_1$ acts as the identity.

Proof. The second statement is a special case of the first one, where the right sequence is empty ($\ell = 0$). Therefore, we only show the first statement.

If we have $M_k \ldots M_1 = N_\ell \ldots N_1$, we can set $p_i = s_{M_i,\mathbf{0}} \in Q_{M_i}$ for all $1 \leq i \leq k$ and $q_j = s_{N_j,\mathbf{0}} \in Q_{N_j}$ for all $1 \leq j \leq \ell$ (where $\mathbf{0} \in \mathbb{Z}^d$ denotes the zero vector). By construction of the Šunić-Ventura automaton, we immediately have that the two state sequences belong to the same element of the automaton semigroup.

For the other direction, we have, for every $1 \leq i \leq k$ and $1 \leq j \leq \ell$, that $p_i = s_{M_i,\boldsymbol{b}_i}$ and $q_j = s_{N_j,\boldsymbol{c}_k}$ where $\boldsymbol{b}_i$ and $\boldsymbol{c}_j$ are vectors form C_{M_i} and C_{N_j}, respectively. Since, by

construction, the action of a state $s_{M,\boldsymbol{b}}$ on the n-adic expansion of a vector is the same as applying $M_{\boldsymbol{b}}$ to the vector, we have

$$(\boldsymbol{b}_k, M_k) \dots (\boldsymbol{b}_1, M_1) = (\boldsymbol{c}_\ell, N_\ell) \dots (\boldsymbol{c}_1, N_1)$$

in $\mathrm{SAff}_d(\mathbb{Z}_n)$ and, thus, from the second components, $M_k \dots M_1 = N_\ell \dots N_1$ as desired. □

Theorem 2.2.2.3 ([3, Theorem 3.7])**.** *The problem* Automaton Group Positive Relation

Input: *a $\mathscr{G}$-automaton $\mathcal{T}$*
Question: *is $\mathscr{P}(\mathcal{T}) \neq \emptyset$?*

is undecidable.

Proof. For reducing Matrix Group Positive Relation to Automaton Group Positive Relation, we map the finite set $\mathcal{M}$ of invertible matrices from $\mathbb{Z}^{4\times4}$ to the Šunić-Ventura automaton $\mathcal{T}_{\mathcal{M},n}$, which is a $\mathscr{G}$-automaton if we choose $n \geq 2$ co-prime to all determinants $\det M$ with $M \in \mathcal{M}$. Clearly, such an n can be computed and the automaton is computable as well since all the transitions of the automaton are of the form

$$s_{M,\boldsymbol{c}} \xrightarrow{x/(\boldsymbol{c}+Mx) \bmod n} s_{M,(\boldsymbol{c}+Mx) \operatorname{div} n}$$

with $x \in \{0, \dots, n-1\}^d$, $M \in \mathcal{M}$ and $\boldsymbol{c} \in C_M$. Finally, by Lemma 2.2.2.2, we have $\mathscr{P}(\mathcal{M}) \neq \emptyset \iff \mathscr{P}(\mathcal{T}_{\mathcal{M},n}) \neq \emptyset$. □

To show that Automaton Group Positive Relation is equivalent to the variants of Automaton Semigroup Neutrality and of Torsion Element where the input automaton is a $\mathscr{G}$-automaton, we use the following fact.

Fact 2.2.2.4. *For a $\mathscr{G}$-automaton $\mathcal{T} = (Q, \Sigma, \delta)$, we have*

$$\begin{aligned}\mathscr{P}(\mathcal{T}) \neq \emptyset &\iff \mathscr{S}(\mathcal{T}) \text{ is a monoid} \\ &\iff \mathscr{S}(\mathcal{T}) \text{ contains an element of torsion}\end{aligned}$$

Proof. If the set of positive relations is non-empty, there is some state sequence $\boldsymbol{q} \in Q^+$ that acts as the identity and is, thus, a neutral element in the semigroup $\mathscr{S}(\mathcal{T})$, which is obviously of torsion.

On the other hand, if $\boldsymbol{q} \in Q^+$ has torsion in $\mathscr{S}(\mathcal{T})$, there are i and j with $i > j$ and $\boldsymbol{q}^i = \boldsymbol{q}^j$ in $\mathscr{S}(\mathcal{T})$. Using the fact, that $\mathcal{T}$ as a $\mathscr{G}$-automaton generates a group, we obtain

$$\boldsymbol{q}^{i-j} \circ u = \boldsymbol{q}^i \bar{\boldsymbol{q}}^j \circ u = \boldsymbol{q}^j \bar{\boldsymbol{q}}^j \circ u = u$$

for all $u \in \Sigma^*$ and, thus, $\boldsymbol{q}^{i-j} =_{\mathcal{T}} \varepsilon$. □

Combining Fact 2.2.2.4 and Theorem 2.2.2.3, we obtain that also our next two problems are undecidable.

Corollary 2.2.2.5. *The problem*

Input: *a $\mathscr{G}$-automaton $\mathcal{T}$*
Question: *does $\mathscr{S}(\mathcal{T})$ contain a neutral element?*

and, thus, Automaton Semigroup Neutrality *are undecidable.*

Corollary 2.2.2.6. *The problem*

Input: *a $\mathscr{G}$-automaton $\mathcal{T}$*
Question: *does $\mathscr{S}(\mathcal{T})$ contain an element of torsion?*

and, thus, Torsion Element *are undecidable.*

To finally reduce Automaton Group Positive Relation to Finite Orbit, we need a few results from Section 1.4.

Proposition 2.2.2.7. *Let $\mathcal{T} = (Q, \Sigma, \delta)$ be a $\mathscr{G}$-automaton. Then, we have:*

$$\mathscr{S}(\mathcal{T}) \text{ contains an element of torsion} \iff \exists \pi \in Q^\omega : |\Sigma^* \circ_\partial \pi| < \infty$$

Proof. If there is some $\boldsymbol{q} \in Q^+$ such that $\boldsymbol{q}$ has torsion in $\mathscr{S}(\mathcal{T})$, then, by Theorem 1.4.2.3, the orbit $\Sigma^* \circ_\partial (\partial \boldsymbol{q})^\omega$ is finite.

On the other hand, if there is some $\pi \in Q^\omega$ with finite orbit $\Sigma^* \circ_\partial \pi$, then there is already some periodic word with a finite orbit by Proposition 1.4.2.5 because the dual $\partial\mathcal{T}$ of the $\mathscr{G}$-automaton $\mathcal{T}$ is a complete and reversible $\mathscr{S}$-automaton. Thus, there is some $\boldsymbol{q} \in Q^+$ with $|\Sigma^* \circ_\partial \boldsymbol{q}^\omega| < \infty$ and Theorem 1.4.2.3 yields that $\partial \boldsymbol{q}$ has torsion in $\mathscr{S}(\mathcal{T})$. □

If we combine Fact 2.2.2.4 and Proposition 2.2.2.7, we obtain that mapping the input $\mathscr{G}$-automaton to its dual is a reduction from Automaton Group Positive Relation to Finite Orbit, which shows that the latter is undecidable.

Corollary 2.2.2.8. *The problem*

Input: *a complete and reversible $\mathscr{S}$-automaton $\mathcal{T} = (Q, \Sigma, \delta)$*
Question: *$\exists \alpha \in \Sigma^\omega : |Q^* \circ \alpha| < \infty$?*

and, thus, Finite Orbit *are undecidable.*

2.2.3 Interlude: the Freeness Problem

The existence of positive relations seems to be connected to the freeness of the generated group. However, the approach given above does not seem to be well-suited to study the freeness problem for automaton groups or semigroups as the Šunić-Ventura automaton will (almost) never generate a free group or semigroup.

The Group Case. If $\mathcal{M}$ is a set of matrices from $\mathbb{Z}^{d\times d}$ with determinant $\det M \in \{-1, 1\}$ for all $M \in \mathcal{M}$, then those matrices generate a group G and $\mathscr{G}(\mathcal{T}_{\mathcal{M},n})$ is (isomorphic to) the semidirect product $\mathbb{Z}^d \rtimes G$ of groups [ŠV12, Lemma 4.1]. Accordingly, the group generated by the Šunić-Ventura automaton contains a non-cyclic abelian subgroup and is, therefore, not free. We will show here that this is also true in the case of general invertible matrices.

Proposition 2.2.3.1 ([6, Proposition A])**.** *Let $\mathcal{M}$ be a non-empty, finite set of matrices $\mathbb{Z}^{d\times d}$ whose determinants are all co-prime to n. Then, $\mathscr{G}(\mathcal{T}_{\mathcal{M},n})$ is free if and only if $d = 1$.*

Proof. First, we consider the case that the group $\mathscr{G}(\mathcal{M})$ generated by the matrices in $\mathcal{M}$ is trivial. This implies that $\mathcal{M}$ can only contain the d dimensional identity matrix (and, thus, in particular, includes the case $d = 1$). In this case, the group generated by $\mathcal{T}_{\mathcal{M},n}$ is (isomorphic to) $\mathbb{Z}^d \rtimes 1$ [ŠV12, Lemma 4.1] and this is free if and only if $d = 1$.

Next, let $\mathscr{G}(\mathcal{M})$ be non-trivial and consider $\mathscr{G}(\mathcal{T}_{\mathcal{M},n})$ as a subgroup G of $\mathbb{Z}_n^d \rtimes \mathscr{G}(\mathcal{M})$. Since we have $\mathbf{0} \in C_M$ for all M, we can consider $\mathscr{G}(\mathcal{M})$ as a subgroup of G. Thus, if $\mathscr{G}(\mathcal{M})$ is not free, G cannot be free either. Therefore, assume that $\mathscr{G}(\mathcal{M})$ is free and contains the non-identity element M. Since we must have $d \geq 2$, there are two distinct elements $\boldsymbol{c}$ and $\boldsymbol{d}$ in C_M (for example, we can take $-\boldsymbol{e}_1$ and $-\boldsymbol{e}_2$ for the first two d dimensional unit vectors $\boldsymbol{e}_1$ and $\boldsymbol{e}_2$). Let H be the subgroup of G generated by $(\boldsymbol{c}, M)$ and $(\boldsymbol{d}, M)$. We will show that H is solvable but not cyclic and, thus, not free, which shows that G cannot be free either.

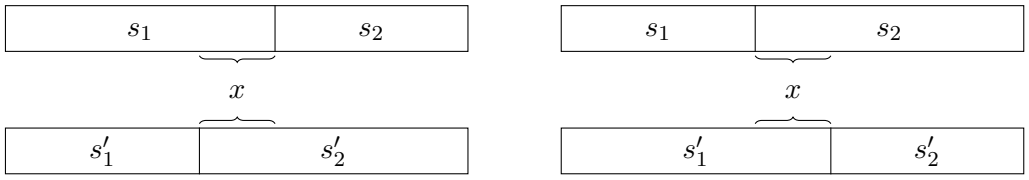
Suppose H is generated by a single element $(\boldsymbol{v}, N)$. Then, there must be $i \neq j$ with $(\boldsymbol{c}, M) = (\boldsymbol{v}, N)^i = (\boldsymbol{c}', N^i)$ and $(\boldsymbol{d}, M) = (\boldsymbol{v}, N)^j = (\boldsymbol{d}', N^j)$. Thus, we must have $N^i = M = N^j$ and, therefore, $N = 1$ since the neutral element is the only element of finite order in the free group $\mathscr{G}(\mathcal{M})$. This is a contradiction, however, since it implies $M = 1$.

To show that H is solvable, we show that the commutator subgroup $[H, H]$ can be considered a subgroup of the abelian (additive) group $\mathbb{Z}_n^d$. Since H is generated by $(\boldsymbol{c}, M)$ and $(\boldsymbol{d}, M)$, every element $h \in H$ can be written as $h = (\boldsymbol{u}, M^i)$ for some i and some $\boldsymbol{u} \in \mathbb{Z}_n^d$. Thus, the commutator of $h = (\boldsymbol{u}, M^i)$ and $k = (\boldsymbol{v}, M^j)$ is $[k, h] = k^{-1}h^{-1}kh = (\boldsymbol{w}, M^{-j}M^{-i}M^jM^i) = (\boldsymbol{w}, 1)$ for some $\boldsymbol{w} \in \mathbb{Z}_n^d$. □

The Semigroup Case. The semigroup case is a bit different to the group case because, here, it does not suffice to show that a subsemigroup is not free. Yet, we can still show that the semigroup generated by a Šunić-Ventura automaton is never free. In fact, we will show a more general algebraic result for subsemigroups of certain semidirect products. However, we first need to discuss some properties of free semigroups.

Remember that we can adjoin a new (disjoint) neutral element $\mathbb{1}$ to every semigroup S and that the resulting semigroup is denoted by $S^{\mathbb{1}}$. Clearly, S is a free semigroup if and only if $S^{\mathbb{1}}$ is a free monoid, which allows us to work with monoids instead of semigroups. A *proper length function* of a monoid M is a homomorphism $\lambda : M \to \mathbb{N}$ such that $\lambda(m) = 0$ implies $m = \mathbb{1}$.[72]

[72] Since $\mathbb{N}$ contains zero, it is the free **monoid** of rank one in additive notation.



(a) s_1 is longer than s_1' (b) s_1 is shorter than s_1'

Figure 2.8: Graphical representation of equidivisibility.

A monoid M is *equidivisable* if, for all $s_1, s_2, s_1', s_2' \in S$ with $s_1 s_2 = s_1' s_2'$, there is some $x \in M$ with $s_1 = s_1' x$ and $x s_2 = s_2'$ or with $s_1 x = s_1'$ and $s_2 = x s_2'$ (see Figure 2.8).

Obviously, every free monoid Σ^* has a natural proper length function that maps a finite word $w \in \Sigma^*$ to its length $|w|$ and is equidivisable by Levi's Lemma [How95, Proposition 7.1.2]. It turns out that the converse also holds [How95, Proposition 7.1.8] and that, thus, these two properties characterize free monoids.

Fact 2.2.3.2. *Let M be a monoid. We have:*

$$M \text{ is a free monoid} \iff M \text{ has a proper length function and is equidivisable}$$

A semigroup S is *right cancellative* if $xs = ys$ implies $x = y$ for all $s, x, y \in S$. Symmetrically, it is *left cancellative* if $sx = sy$ implies $x = y$ for all $s, x, y \in S$ and it is *cancellative* if it is both left and right cancellative. Clearly, every free monoid is cancellative.

The semigroup generated by a Šunić-Ventura automaton for a set $\mathcal{M}$ of matrices from $\mathbb{Z}^{d\times d}$ is (isomorphic to) a subsemigroup of the affine semigroup $\mathrm{SAff}_d(\mathbb{Z}) = \mathbb{Z}^d \rtimes \mathrm{SGL}_d(\mathbb{Z})$. This semidirect product has a crucial property: the action of $\mathrm{SGL}_d(\mathbb{Z})$ on $\mathbb{Z}^d$ is compatible with the monoid structure of $\mathbb{Z}^d$.[73] In general, a semigroup S *acts on a monoid* M (from the left) via a homomorphism $\alpha : S \to \operatorname{End} M, s \mapsto \alpha_s$ where $\operatorname{End} M$ is the monoid of **monoid** endomorphisms of M, i. e. we have $\alpha_s(\mathbb{1}) = \mathbb{1}$ for all $s \in S$. If we have such an action and we form the semidirect product $M \rtimes S$, we have $(\mathbb{1}, s)(\mathbb{1}, t) = (\mathbb{1}\alpha_s(\mathbb{1}), st) = (\mathbb{1}, st)$. Thus, by identifying s with $(\mathbb{1}, s)$, we obtain S as a subsemigroup of $M \rtimes S$.

[73] Actually, it is also compatibles with the inverses but we do not need this property in the following.

We will proceed by showing that certain subsemigroups of $M \rtimes S$ are not free. Afterwards, we will apply these general results to the specific setting of the affine semigroup and Šunić-Ventura automata.

Lemma 2.2.3.3 ([6, Lemma D])**.** *Let a semigroup S act on a monoid M (from the left) and let T be a subsemigroup of $M \rtimes S$ such that S is in turn a subsemigroup of T. Then, we have:*

$$T \text{ is a free semigroup} \implies S \text{ is a free semigroup}$$

Proof. Let T be a free semigroup or, equivalently, let $T^{\mathbb{1}}$ be a free monoid. This implies that $T^{\mathbb{1}}$ has a proper length function and is equidivisable. We will show that $S^{\mathbb{1}}$ inherits both these properties and is, thus, a free monoid (by Fact 2.2.3.2), which shows that S is a free semigroup.

We define the length of $\mathbb{1} \in S^{\mathbb{1}}$ as 0 and the length of $s \in S$ to be the same as the length of $(\mathbb{1}, s)$ in $T^{\mathbb{1}}$, which clearly gives us a proper length function. To show that $S^{\mathbb{1}}$ is equidivisable, let $s_1 s_2 = s_1' s_2'$ for $s_1, s_2, s_1', s_2' \in S^{\mathbb{1}}$. We have to show that there is some $x \in S^{\mathbb{1}}$ with $s_1 = s_1' x$ and $x s_2 = s_2'$ or with $s_1 x = s_1'$ and $s_2 = x s_2'$ (again, see Figure 2.8).

If we have $s_1 = \mathbb{1}$, we can set $x = s_1'$ as we, then, have $s_1 x = s_1 s_1' = \mathbb{1} s_1' = s_1'$ and $s_2 = \mathbb{1} s_2 = s_1 s_2 = s_1' s_2' = x s_2'$. The cases $s_1' = \mathbb{1}$, $s_2 = \mathbb{1}$ and $s_2' = \mathbb{1}$ can be handled symmetrically. Thus, let $s_1, s_2, s_1', s_2' \in S = S^{\mathbb{1}} \setminus \{\mathbb{1}\}$. Because S is a subsemigroup of T, we have

$$(\mathbb{1}, s_1)(\mathbb{1}, s_2) = (\mathbb{1}, s_1')(\mathbb{1}, s_2')$$

in T and, therefore, also in $T^{\mathbb{1}}$. Because $T^{\mathbb{1}}$ is equidivisable, there is some $y \in T^{\mathbb{1}}$ with $(\mathbb{1}, s_1) = (\mathbb{1}, s_1')y$ and $y(\mathbb{1}, s_2) = (\mathbb{1}, s_2')$ or with $(\mathbb{1}, s_1)y = (\mathbb{1}, s_1')$ and $(\mathbb{1}, s_2) = y(\mathbb{1}, s_2')$. If $y = \mathbb{1}$, we obtain $s_1 = s_1'$ and $s_2 = s_2'$ from the second components (in both cases) and can set $x = \mathbb{1}$. Otherwise, y must be of the form $y = (m, x) \in T = T^{\mathbb{1}} \setminus \{\mathbb{1}\}$, which yields the sought element x. That x indeed satisfies the required condition can be seen by restricting the equations to their second components (in both cases). □

Proposition 2.2.3.4 ([6, Proposition E])**.** *Let $\alpha : S \to \mathrm{End}(M), s \mapsto \alpha_s$ be an action of a semigroup S on a monoid M and let T be a subsemigroup of $M \rtimes S$ such that T contains S as a subsemigroup. If there is some $(m, s) \in T$ with $m \neq \mathbb{1}$ (i. e. if S is a proper subsemigroup) and we have $(\alpha_s(m), s) \in T$ as well, then T cannot be a free semigroup.*

Proof. Suppose we have $(m, s), (m', s) \in T$ for some $m \neq \mathbb{1}$ and $m' = \alpha_s(m)$. We show the statement by contradiction and assume that T is a free semigroup or, equivalently, that $T^{\mathbb{1}}$ is a free monoid.

This implies that $T^{\mathbb{1}}$ is equidivisable. In particular, since we have

$$(\mathbb{1}, s)(m, s) = (\mathbb{1}m', s^2) = (m', s)(\mathbb{1}, s)$$

in T (and, therefore, also in $T^{\mathbb{1}}$), there is some $y \in T^{\mathbb{1}}$ with $y(m, s) = (\mathbb{1}, s)$ or with $(m, s) = y(\mathbb{1}, s)$.[74] In both cases, we cannot have $y = \mathbb{1}$ because we have $m \neq \mathbb{1}$. Thus, y is of the form $y = (n, x)$ for some $n \in M$ and $x \in S$. From the second component of the respective equation, we obtain $xs = s$, again, in both cases.

Finally, since we assumed T to be a free semigroup, S must be a free semigroup as well by Lemma 2.2.3.3 and we obtain that the (thus) free monoid $S^{\mathbb{1}}$ is cancellative. This yields $x = \mathbb{1}$, which constitutes a contradiction. □

[74] Of course, we have another equation in both cases but we will not need it.

Counter Example 2.2.3.5. It might be tempting to assume that no subsemigroup of $M \rtimes S$ which contains S as a proper subsemigroup is a free semigroup. However, this is not true.

For a counter example,[75] let $S = q^+$ act on the monoid $M = \{a, b\}^*$ via the action α induced by $\alpha_q(a) = a$ and $\alpha_q(b) = ab$ and consider the subsemigroup T generated by (ε, q) and (b, q) in $M \rtimes S$. Clearly, S is a proper subsemigroup of T. However, T is freely generated by (ε, q) and (b, q) (and, thus, a free semigroup). We will show that every element $t = (w, q^i) \in T$ has a unique factorization over these generators. We observe

[75] The current author would like to thank Armin Weiß for pointing out this counter example.

that all elements in $(\varepsilon, q)T^{\mathbb{1}} = \{(\varepsilon, q)t' \mid t' \in T^{\mathbb{1}}\}$ either have an empty first component or one which starts with an a. On the other hand, the first component of all elements in $(b, q)T^{\mathbb{1}}$ must start with a b. Thus, the first letter of w uniquely determines whether t is in $(\varepsilon, q)T^{\mathbb{1}}$ or in $(b, q)T^{\mathbb{1}}$ and our claim follows by induction.

We can use Proposition 2.2.3.4 to handle the non-trivial cases in the proof that no Šunić-Ventura automaton can generate a free semigroup.

Theorem 2.2.3.6 ([6, Proposition B])**.** *Let $\mathcal{M}$ be a non-empty, finite set of matrices from $\mathbb{Z}^{d\times d}$ and let $n \geq 2$. Then, $\mathscr{S}(\mathcal{T}_{\mathcal{M},n})$ is never a free semigroup.*

Proof. If $\mathcal{M}$ only consists of the zero matrix 0 from $\mathbb{Z}^{d\times d}$, then the semigroup generated by $\mathcal{T}_{\mathcal{M},n}$ consists of a single element (as C_0 only contains the d dimensional zero vector) and is, thus, not a free semigroup.

Otherwise, there is some $M \in \mathcal{M}$ with maximum absolute row sum norm $\|M\| \geq 1$. Thus, we have $-\boldsymbol{e}_1 \in C_M$ where $\boldsymbol{e}_1$ is the first unit vector of the canonical base for $\mathbb{Z}^d$. The i^{th} component of $\boldsymbol{v} = M(-\boldsymbol{e}_1)$ is $-m_{i,1}$, the entry in the first column and i^{th} row of M. Since we have $m_{i,1} \leq \|M\|$, we obtain $\boldsymbol{v} \in C_M$ as well. Thus, if we consider $\mathscr{S}(\mathcal{T}_{\mathcal{M},n})$ as a subsemigroup T of $\mathrm{SAff}_d(\mathbb{Z}) = \mathbb{Z}^d \rtimes \mathrm{SGL}_d(\mathcal{M})$, we have $(-\boldsymbol{e}_1, M) \in T$ and $(M(-\boldsymbol{e}_1), M) \in T$. Additionally, $\mathrm{SGL}_d(\mathcal{M})$ is a subsemigroup of T because C_M always contains the d dimensional zero vector. This allows us to apply Proposition 2.2.3.4 to conclude the proof. □

2.3 Finiteness Problem

The finiteness problem for automaton semigroups AUTOMATON SEMIGROUP FINTENESS

Input: an $\mathscr{S}$-automaton $\mathcal{T}$
Question: is $\mathscr{S}(\mathcal{T})$ finite?

was shown to be undecidable by Gillibert [Gil14]. On the other hand, the corresponding problems for inverse automaton semigroups and automaton groups remain open.[76] In this section, we will strengthen Gillibert's result and show that the problem BI-REVERSIBLE, INVERTIBLE AUTOMATON SEMIGROUP FINITENESS

Input: a bi-reversible, invertible (possibly partial) $\mathscr{S}$-automaton $\mathcal{T}$
(i. e. a bi-reversible $\overline{\mathscr{S}}$-automaton)
Question: is $\mathscr{S}(\mathcal{T})$ finite?

is undecidable. Just like Gillibert's result, our proof is based on Wang tilings but, while Gillibert completed the resulting automaton by adding a sink state (compare to Corollary 1.1.1.3), we will work with partial automata instead.

This result is a step closer to showing the undecidability of the finiteness problem for inverse automaton semigroups INVERSE AUTOMATON SEMIGROUP FINITENESS:

Input: an invertible (possibly partial) $\mathscr{S}$-automaton $\mathcal{T}$
(i. e. an $\overline{\mathscr{S}}$-automaton)
Question: is $\overline{\mathscr{S}}(\mathcal{T})$ finite?

[76] For the group case, see [GNS00, 7.2 b)].

Similar to the word problem (see Section 2.1), solving the version for inverse automaton semigroups could be interesting as a step stone to the finiteness problem for automaton groups AUTOMATON GROUP FINITENESS

Input: an invertible, complete $\mathscr{S}$-automaton $\mathcal{T}$ (i. e. a $\mathscr{G}$-automaton)
Question: is $\mathscr{G}(\mathcal{T})$ finite?

Unfortunately, our result here shows neither and we will discuss the connections between these problems at the end of this section. However, we will give a partial solution by showing that a generalized version of the finiteness problem for automaton groups is undecidable.[77] We will re-formulate this generalized version using results from Section 1.4 as the problem to find an ω-word with a given finite prefix and an infinite orbit and as the finiteness problem for left principal ideals in a semigroup generated by a complete and reversible automaton.

[77] Later on, in Subsection 2.4.3, we will re-visit the finiteness problem for automaton groups and show that it is decidable for the sub-class of automata with bounded activity.

Attribution. The results on the bi-reversible, partial case at the beginning of this section are joint work with Daniele D'Angeli and Emanuele Rodaro [3]. Similar to Gillibert's original approach [Gil14], the undecidability proof is based on an undecidability result for Wang tilings, namely a result by Lukkarila showing that the tiling problem remains undecidable for 4-way-deterministic Wang tile sets [Luk09]. This result, in turn, uses the existence of a 4-way-deterministic Wang tile set that can be mapped homomorphically to Robinson's aperiodic tile set [Rob71], which was found by Kari and Papasoglu [KP99]. We can simplify the original proof (from [3]) slightly by using the results on infinite orbits from Section 1.4.

Finally, the partial solution to the finiteness problem for automaton groups given at the end of this section and the two alternative formulations are joint work not only with Daniele D'Angeli and Emanuele Rodaro but also with Dominik Francoeur [8]. The construction heavily relies on the one given by Gillibert to show that the order problem for automaton groups is undecidable [Gil18].

2.3.1 An Undecidability Result Due to Lukkarila

Our proof for extending Gillibert's undecidability result to bi-reversible partial automata is – just like Gillibert's original proof – heavily based on Wang tiles. Therefore, we start our discussion by introducing a result by Lukkarila in this area.

Wang Tilings. Let C be a finite set of *colors*. A *Wang tile* over C is a quadruple $t = (t_N, t_W, t_S, t_E) \in C^4$, for which we also use the more graphical notation $t = \begin{smallmatrix} & c_N & \\ c_W & & c_E \\ & c_S & \end{smallmatrix}$ in this context. For $D \in \{N, W, S, E\}$, we say that t_D is the color at the D-edge.[78] A *tile set* $\mathcal{W}$ is a finite set of Wang tiles. It is *CD-deterministic* for $CD \in \{NW, SW, SE, NE\}$ if a tile $t \in \mathcal{W}$ is uniquely determined by the colors at its C- and D-edges. It is *4-way-deterministic* if it is CD-deterministic for all $CD \in \{NW, SW, SE, NE\}$.

[78] N, W, S and E obviously stand for "north", "west", "south" and "east" here.

A *tiling of the discrete plane* $\mathbb{Z}^2$, or $\mathbb{Z}^2$-tiling for short, for a tile set $\mathcal{W}$ is a map $f : \mathbb{Z}^2 \to \mathcal{W}$ such that adjacent tiles share the same color on their common edge, i. e. we

have $f(x,y)_E = f(x+1,y)_W$ and $f(x,y)_N = f(x,y+1)_S$ for all $x, y \in \mathbb{Z}$. Analogously, a *tiling of the first quadrant* $\mathbb{N}^2$, or $\mathbb{N}^2$-tiling for short, for $\mathcal{W}$ is a map $f : \mathbb{N}^2 \to \mathcal{W}$ with $f(x,y)_E = f(x+1,y)_W$ and $f(x,y)_N = f(x,y+1)_S$ for all $x, y \in \mathbb{N}$. The Wang tile set $\mathcal{W}$ is said to *tile* the discrete plane $\mathbb{Z}^2$ (or the first quadrant $\mathbb{N}^2$, respectively) if there is a $\mathbb{Z}^2$-tiling (an $\mathbb{N}^2$-tiling, respectively) for $\mathcal{W}$.

The following fact is well-known.

Fact 2.3.1.1. *A tile set $\mathcal{W}$ tiles the discrete plane $\mathbb{Z}^2$ if and only if it tiles the first quadrant $\mathbb{N}^2$.*

Periodicity. For an $\mathbb{N}^2$-tiling $f : \mathbb{N}^2 \to \mathcal{W}$ for a tile set $\mathcal{W} \subseteq C^4$ with the colors C, we can define the notion of a *horizontal word*: the y^{th} horizontal word is

$$f(0,y)_S f(1,y)_S f(2,y)_S \ldots \in C^\omega.$$

An $\mathbb{N}^2$-tiling is *non-y-recurrent* if all its horizontal words are pairwise distinct; otherwise, it is *y-recurrent*.

A $\mathbb{Z}^2$-tiling f is called *periodic* if there is a (non-zero) *periodicity vector* $\boldsymbol{v} \in \mathbb{Z}^2$ with $f(\boldsymbol{p}+\boldsymbol{v}) = f(\boldsymbol{p})$ for all $\boldsymbol{p} \in \mathbb{Z}^2$; otherwise, it is called *non-periodic*. If there is a periodicity vector of the form $(x,0)$ for some $x \in \mathbb{Z}$ or of the form $(0,y)$ for some $y \in \mathbb{Z}$, then the $\mathbb{Z}^2$-tiling f is called *horizontally periodic* or *vertically periodic*, respectively. A tile set is *aperiodic* if it admits a non-periodic $\mathbb{Z}^2$ tiling but does not admit a periodic one.

The following connection between periodicity and horizontal and vertical periodicity is well-known (see e. g. [Rob71, §1. Introduction]).

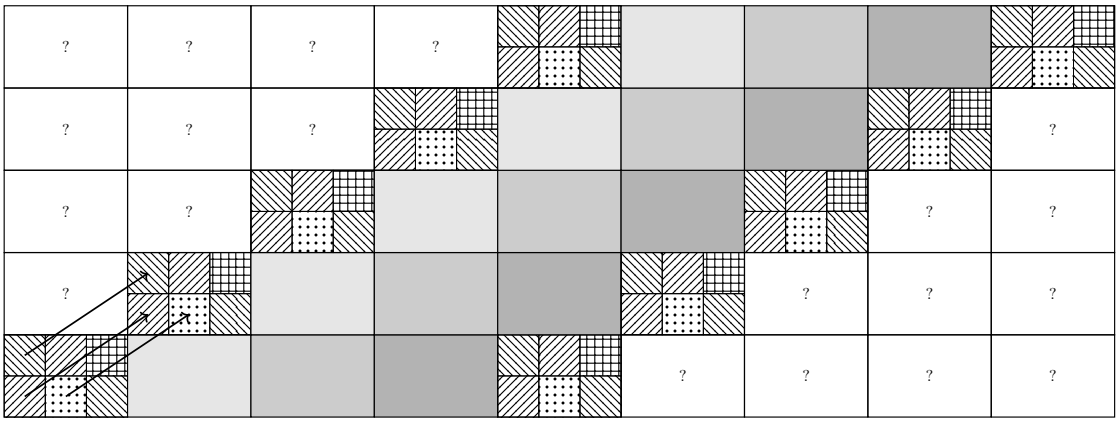
Fact 2.3.1.2. *A tile set $\mathcal{W}$ admits a periodic $\mathbb{Z}^2$-tiling if and only it admits a $\mathbb{Z}^2$-tiling which is horizontally and vertically periodic.*

Proof idea. One implication is obvious. The idea for the other direction is best understood by considering Figure 2.9. We start with an arbitrary periodic $\mathbb{Z}^2$-tiling and combine its tiles into blocks whose size is given by the periodicity vector. As these blocks are of finite size, there are only finitely many possibilities for them. Therefore, there must be a block which appears twice in the same row (see Figure 2.9a). This yields a pattern which we can repeat to tile the whole plane. The resulting $\mathbb{Z}^2$-tiling has a horizontal as well as a vertical period (see Figure 2.9b). □

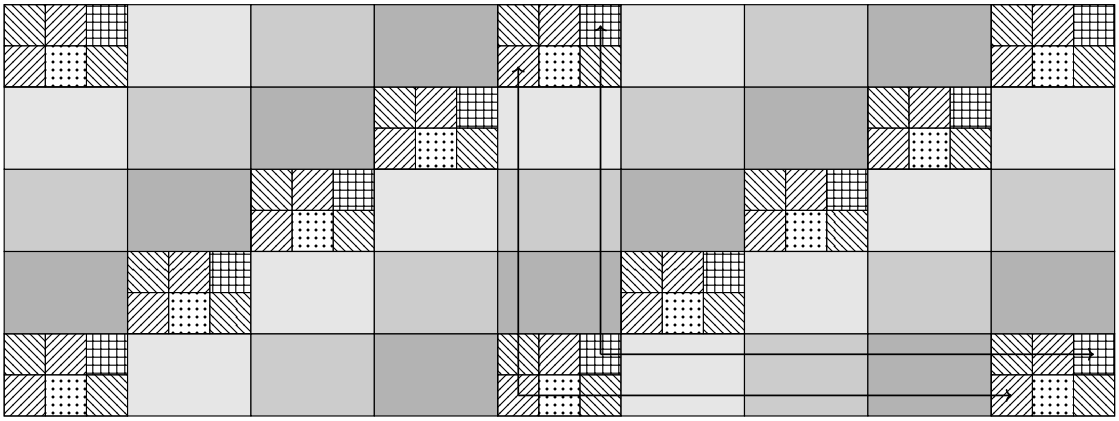
Using very similar ideas, we can also link the periodicity of $\mathbb{Z}^2$-tilings to the non-y-recurrence of $\mathbb{N}^2$-tilings.

Lemma 2.3.1.3 ([3, Lemma 4.4])**.** *A tile set $\mathcal{W}$ tiles the first quadrant in a y-recurrent way if and only if it tiles the discrete plane periodically.*

Proof. Let f be a y-recurrent $\mathbb{N}^2$-tiling for $\mathcal{W}$ such that the i^{th} and the j^{th} horizontal words coincide for $i < j$ (see Figure 2.10). Because there are only finitely many possibilities, there have to be $x_1 < x_2$ such that $f(x_1, y) = f(x_2, y)$ for all $y \in \{i, \ldots, j-1\}$. Therefore,



(a) A tiling with periodicity vector $(3, 2)$. Blocks of the same shade contain the same tiles.



(b) Repeating the pattern yields both a horizontal and a vertical period.

Figure 2.9: Periodic tilings

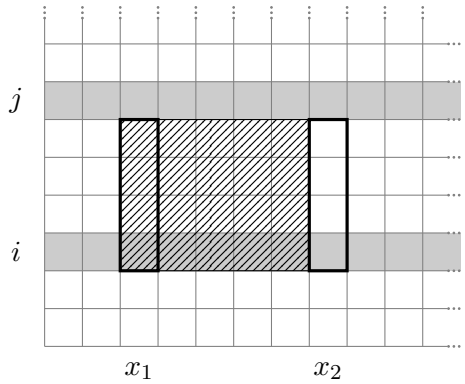


Figure 2.10: A y-recurrent $\mathbb{N}^2$-tiling

we can repeat the pattern given by the tiles at positions $\{x_1, \dots, x_2 - 1\} \times \{i, \dots, j - 1\}$ (i. e. the hatched part in Figure 2.10) infinitely often in both directions, which yields a periodic $\mathbb{Z}^2$-tiling.

If, on the other hand, we have a periodic $\mathbb{Z}^2$-tiling, then, by Fact 2.3.1.2, we also have a vertically periodic $\mathbb{Z}^2$-tiling. Restricting this tiling into a map $\mathbb{N}^2 \to \mathcal{W}$ yields a y-recurrent $\mathbb{N}^2$-tiling. □

A *homomorphism of tile sets* is a map $\varphi : \mathcal{V} \to \mathcal{W}$ from a tile set $\mathcal{V}$ to a tile set $\mathcal{W}$ such that $t_D = t'_D$ implies $\varphi(t)_D = \varphi(t')_D$ for all $t \in \mathcal{V}$ and all $D \in \{N, W, S, E\}$. Clearly, a periodic $\mathbb{Z}^2$-tiling for $\mathcal{V}$ induces a periodic $\mathbb{Z}^2$-tiling for $\mathcal{W}$. Thus, if $\mathcal{W}$ is aperiodic and $\mathcal{V}$ admits any $\mathbb{Z}^2$-tiling, then $\mathcal{V}$ is aperiodic as well. A tile set is *homomorphically aperiodic* if it can be mapped homomorphically to an aperiodic tile set.

Undecidability. It was shown by Berger [Ber66] that the tiling problem for Wang tiles

Input: a tile set $\mathcal{W}$
Question: does $\mathcal{W}$ tile the discrete plane $\mathbb{Z}^2$?

is undecidable. Later, the proof was simplified by Robinson [Rob71]. Crucial to both undecidability proofs – in fact, to any such proof – is the existence of an aperiodic tile set. Using an aperiodic 4-way-deterministic tile set constructed by Kari and Papasoglu [KP99], Lukkarila [Luk09] could show the undecidability of the problem 4-Way-Deterministic Tiling

Input: a 4-way-deterministic, homomorphically aperiodic tile set $\mathcal{W}$
Question: does $\mathcal{W}$ tile the discrete plane $\mathbb{Z}^2$?

We will reduce[79] 4-Way-Deterministic Tiling to Bi-Reversible, Invertible Automaton Semigroup Finiteness to show the undecidability of the latter.

[79] Strictly speaking, we will give a co-reduction again.

2.3.2 The Reduction

There is a very natural connection between Wang tile sets and automata. For a Wang tile set $\mathcal{W} \subseteq C^4$, we define the (partial) automaton $\mathcal{T}(\mathcal{W}) = (C, C, \delta)$ by

$$\delta = \left\{ c_W \xrightarrow{c_S/c_N} c_E \;\middle|\; \begin{smallmatrix} \square & c_N & \square \\ c_W & & c_E \\ \square & c_S & \square \end{smallmatrix} \in \mathcal{W} \right\}.$$

Clearly, for a given tile set $\mathcal{W}$, the automaton $\mathcal{T}(\mathcal{W})$ can be computed. Furthermore, we have the following straight-forward connections.

Fact 2.3.2.1. *Let $\mathcal{W}$ be a Wang tile set. We have:*

- *$\mathcal{T}(\mathcal{W})$ is deterministic $\iff$ $\mathcal{W}$ is SW-deterministic,*
- *$\mathcal{T}(\mathcal{W})$ is invertible $\iff$ $\mathcal{W}$ is NW-deterministic,*
- *$\mathcal{T}(\mathcal{W})$ is inverse-reversible $\iff$ $\mathcal{W}$ is NE-deterministic and*
- *$\mathcal{T}(\mathcal{W})$ is reversible $\iff$ $\mathcal{W}$ is SE-deterministic.*

Additionally, we can link the properties that a tile set tiles the discrete plane and that the associated automaton generates an infinite semigroup.

Lemma 2.3.2.2. *Let $\mathcal{W}$ be a homomorphically aperiodic, SW-deterministic tile set. Then, we have:*

$$\mathcal{W} \text{ tiles the discrete plane } \mathbb{Z}^2 \iff |\mathscr{S}(\mathcal{T}(\mathcal{W}))| = \infty$$

Proof. Let C be the colors belonging to $\mathcal{W}$. If $\mathcal{W}$ tiles the discrete plane $\mathbb{Z}^2$, then, in particular, it also tiles the first quadrant $\mathbb{N}^2$ and this $\mathbb{N}^2$-tiling must be non-y-recurrent since a y-recurrent $\mathbb{N}^2$-tiling yields a periodic $\mathbb{Z}^2$-tiling by Lemma 2.3.1.3, which cannot exist as $\mathcal{W}$ is homomorphically aperiodic. If we denote the i^{th} horizontal word of this non-y-recurrent $\mathbb{N}^2$-tiling by α_i, then $\alpha_0, \alpha_1, \ldots$ are the nodes of an infinite path in the orbital graph $C^* \circ \alpha_0$ with respect to $\mathcal{T}(\mathcal{W})$ (whose labels are given by the colors on the west side). This shows that $\mathscr{S}(\mathcal{T}(\mathcal{W}))$ must be infinite.

For the other direction, suppose that $\mathscr{S}(\mathcal{T}(\mathcal{W}))$ is infinite. Then, by Corollary 1.4.1.14, there is some ω-word α with an infinite orbit under the action of $\mathcal{T}(\mathcal{W})$. Thus, its orbital graph $C^* \circ \alpha$ contains an infinite simple path

$$\alpha = \alpha_0 \xrightarrow{c_1} \alpha_1 \xrightarrow{c_2} \alpha_2 \xrightarrow{c_3} \cdots$$

since there are infinitely many nodes reachable from α and the out-degree of all nodes is bounded by $|C|$. This path (uniquely) induces a (non-y-recurrent) $\mathbb{N}^2$-tiling whose i^{th} horizontal word is given by α_i and whose west side is labeled by $c_1 c_2 \ldots$. By Fact 2.3.1.1, this also yields a $\mathbb{Z}^2$-tiling (which has to be aperiodic since $\mathcal{W}$ is homomorphically aperiodic). □

From the above lemma, we obtain the undecidability of the strengthened version of the finiteness problem for automaton semigroups.

Theorem 2.3.2.3. *The problem* Bi-Reversible, Invertible Automaton Semigroup Finiteness

Input:	*a bi-reversible, invertible (possibly partial) $\mathscr{S}$-automaton $\mathcal{T}$ (i. e. a bi-reversible $\overline{\mathscr{S}}$-automaton)*
Question:	*is $\mathscr{S}(\mathcal{T})$ finite?*

is undecidable.

Proof. We give a co-reduction from 4-Way-Deterministic Tiling to the stated problem. We map the 4-way-deterministic, homomorphically aperiodic tile set $\mathcal{W}$ to the (computable) automaton $\mathcal{T}(\mathcal{W})$, which is an invertible, bi-reversible $\mathscr{S}$-automaton by Fact 2.3.2.1. The correctness of this co-reduction follows immediately from Lemma 2.3.2.2. □

2.3.3 The Case of Invertible Automaton Structures

Inverse Automaton Semigroups. In the group case, we have the well-known connection from Fact 1.4.1.15, which states that a $\mathscr{G}$-automaton generates an infinite group if and only if it generates an infinite semigroup.

In fact, we have proved this connection using Fact 0.3.2.2, which yields that, for a $\mathscr{G}$-automaton $\mathcal{T} = (Q, \Sigma, \delta)$ and all $w \in Q^\infty$, the positive orbit $Q^* \circ w$ is of the same size as $\widetilde{Q}^* \circ w$, where we also consider inverses.

However, the corresponding result for the orbits of (non-complete) $\overline{\mathscr{S}}$-automata does not hold (see Counter-Example 0.3.2.3). Thus, it might not be too surprising that the inverse semigroup generated by an $\overline{\mathscr{S}}$-automaton can be infinite while the generated semigroup is finite.

Counter Example 2.3.3.1 (see also [3, Lemma 2.6])**.** We use the same idea as in Counter-Example 0.3.2.3. Let $\mathcal{T} = (Q, \Sigma, \delta)$ be an arbitrary $\mathscr{G}$-automaton generating an infinite group such that there is a state q_1 whose action is the identity on Σ^*. Let $\widehat{\Sigma}$ be a disjoint copy of Σ and define $\mathcal{T}' = (Q, \Sigma \uplus \widehat{\Sigma}, \delta')$ by

$$\delta' = \{p \xrightarrow{a/\hat{b}} q \mid p \xrightarrow{a/b} q \in \delta\}.$$

Then, $\mathscr{S}(\mathcal{T}')$ as well as $\mathscr{S}(\overline{\mathcal{T}'})$ are finite while $\overline{\mathscr{S}}(\mathcal{T}')$ is infinite.

The former is obvious since the partial action of any state sequence containing more than one state is undefined on every non-empty word. For the latter, we need to observe that the partial action of q_1 in $\mathcal{T}'$ consists of mapping a to $\hat{a}$ for every a. Accordingly, the partial action of $\overline{q_1}$ in $\overline{\mathcal{T}'}$ removes the hat decoration. Thus, the partial action of $q \in Q$ in $\mathcal{T}$ is the same as that of $\overline{q_1} q$ in $\mathcal{T}'$.

This prevents us from directly using the above reduction to show that Inverse Automaton Semigroup Finiteness is undecidable. The problem here is that, if $\overline{\mathscr{S}}(\mathcal{T}(\mathcal{W}))$ is infinite, then we only know that we have an ω-word with an infinite orbit under the combined partial actions of the states and their inverses. On the Wang tile side, this corresponds to also allowing tiles mirrored at the horizontal axis. However, if we have such tiles, then we can tile the discrete plane $\mathbb{Z}^2$ if we can tile a single horizontal line and the corresponding problem is suddenly decidable.

Automaton Groups. That the resulting automaton is partial is somewhat intrinsic to the above construction.[80] Clearly, if $\mathcal{T}(\mathcal{W})$ is complete for some tile set $\mathcal{W}$, then we can find a Wang tile $t \in \mathcal{W}$ with $t_S = c_S$ and $t_W = c_W$ for any pair c_S, c_W of colors, which implies that $\mathcal{W}$ tiles the discrete plane $\mathbb{Z}^2$.

However, we can show a partial result for Automaton Group Finitness using the construction given by Gillibert to show that there an automaton group with an undecidable order problem:[81]

Constant:	a $\mathscr{G}$-automaton $\mathcal{T} = (Q, \Sigma, \delta)$
Input:	a finite state sequence $\boldsymbol{q} \in Q^*$
Question:	has $\boldsymbol{q}$ finite order in $\mathscr{G}(\mathcal{T})$?

[80] In fact, Gillibert's proof for the undecidability of Automaton Semigroup Finiteness [Gil14] also seems to have a "partial nature". However, he made the automaton complete by adding a sink state (compare to Corollary 1.1.1.3).

[81] This construction is entirely different to the one for the finiteness problem that was mentioned above.

Theorem 2.3.3.2 ([8, Theorem 3.1])**.** *The decision problem*[82]

Constant: *a $\mathscr{G}$-automaton $\mathcal{T} = (Q, \Sigma, \delta)$*
Input: *a finite word $w \in \Sigma^*$*
Question: *is $\mathscr{G}(\mathcal{T}) \cdot w = \{g \cdot w \mid g \in \mathscr{G}(\mathcal{T})\}$ finite?*

is undecidable for some $\mathscr{G}$-automaton $\mathcal{T}$.

[82] $\mathscr{G}(\mathcal{T}) \cdot w$ is well-defined by Fact 0.3.1.1.

Proof (adapted from the proof of [8, Theorem 3.1]). Although it is not explicitly stated in his proof, Gillibert actually shows the undecidability of the decision problem

Constant: a $\mathscr{G}$-automaton $\mathcal{R} = (P, \Gamma, \tau)$ and a state $\$ \in P$
Input: a finite sequence $\boldsymbol{p} \in P^*$ of states
Question: has $\$\Lambda(\boldsymbol{p})$ finite order in $\mathscr{G}(\mathcal{R})$?

where $\Lambda : P^* \to P^*$ is given by $\Lambda(\varepsilon) = \varepsilon$ and $\Lambda(\hat{p}\boldsymbol{p}) = \Lambda(\boldsymbol{p})\hat{p}\Lambda(\boldsymbol{p})$ [Gil18].[83]

[83] In Gillibert's paper, the function is called Q, actually. However, this notation clashes with the convention of using Q to denote the state sets of automata followed in this work. Therefore, we use Λ instead. Additionally, Gillibert uses right actions to define automaton groups. Therefore, we mirror the ordering here.

We take the $\mathscr{G}$-automaton $\mathcal{R}$ and extend it into a new $\mathscr{G}$-automaton $\mathcal{T} = (Q, \Sigma, \delta)$. Then, we reduce the above version of the order problem of $\mathcal{R}$ to the generalized finiteness problem for $\mathcal{T}$ from the theorem statement.

As the alphabet of $\mathcal{T}$, we use $\Sigma = \Gamma \uplus \{a_p \mid p \in P\} \times \{0,1\} \uplus \{*, \#\}$, i. e. we add two new special letters $*$ and $\#$ as well as two new letters $(a_p, 0)$ and $(a_p, 1)$ for every state $p \in P$. Similarly, we use $Q = P \uplus \{s, t, \mathrm{id}\} \uplus \{\#_p \mid p \in P\}$ for the state set, i. e. we add three new states s, t and id as well as a new state $\#_p$ for every old state $p \in P$. Of course, we also add new transitions

$$\begin{aligned}\delta' = \tau &\cup \{s \xrightarrow{*/*} t, t \xrightarrow{\#/\#} \$\} \cup \{t \xrightarrow{(a_p,1)/(a_p,0)} t, t \xrightarrow{(a_p,0)/(a_p,1)} \#_p \mid p \in P\} \\ &\cup \{\#_p \xrightarrow{(a_q,i)/(a_q,i)} \#_p, \#_p \xrightarrow{\#/\#} p \mid p, q \in P, i \in \{0,1\}\} \\ &\cup \{\mathrm{id} \xrightarrow{a/a} \mathrm{id} \mid a \in \Sigma\},\end{aligned}$$

which are depicted schematically in Figure 2.11, and make the automaton complete by adding a transition to the identity state whenever some transition is missing:

$$\delta = \delta' \cup \{q \xrightarrow{a/a} \mathrm{id} \mid q \in Q, a \in \Sigma, \nexists a' \in \Sigma, q' \in Q : q \xrightarrow{a/a'} q' \in \delta'\}$$

Note that the resulting automaton is indeed a $\mathscr{G}$-automaton!

For the reduction of the strengthened version of the order problem to the generalized version of the finiteness problem, we map the input sequence $\boldsymbol{p} = p_\ell \dots p_1$ to the finite word $w = *w' = *(a_{p_1}, 0) \dots (a_{p_\ell}, 0)\#$, which is obviously computable. In the remainder of this proof, we show that $\$\Lambda(\boldsymbol{p})$ has finite order in $\mathscr{G}(\mathcal{R})$ if and only if $\mathscr{G}(\mathcal{T}) \cdot w$ is finite.

First, we show that $\$\Lambda(\boldsymbol{p})$ has finite order in $\mathscr{G}(\mathcal{R})$ if and only if it has in $\mathscr{G}(\mathcal{T})$. We do this, by showing that $(\$\Lambda(\boldsymbol{p}))^i$ and $(\$\Lambda(\boldsymbol{p}))^j$ are distinct in $\mathscr{G}(\mathcal{R})$ if and only if they are distinct in $\mathscr{G}(\mathcal{T})$ for $i, j \in \mathbb{N}$. If they are distinct in $\mathscr{G}(\mathcal{R})$, there is some witness $u \in \Gamma^*$ which they act differently on. Since we have $\tau \subseteq \delta' \subseteq \delta$, this is also a witness for their difference in $\mathscr{G}(\mathcal{T})$. For the other direction, suppose that $(\$\Lambda(\boldsymbol{p}))^i$ is different to $(\$\Lambda(\boldsymbol{p}))^j$ in $\mathscr{G}(\mathcal{T})$. Then, there must be some witness $u \in \Sigma^*$ which they act differently on. We are done if u is already in Γ^*. Otherwise, we can factorize $u = u_1 a u_2$ with $u_1 \in \Gamma^*$, $a \in \Sigma \setminus \Gamma$ and $u_2 \in \Sigma^*$. By the construction of $\mathcal{T}$, we remain in states from P if we start in P and

read letters from Γ. If we read a letter from $\Sigma \setminus \Gamma$, we go to id, which yields the cross diagrams

$$(\$\Lambda(\boldsymbol{p}))^i \begin{array}{c} u_1 \\ \downarrow \\ v_1 \end{array} \begin{array}{c} a \\ \downarrow \\ a \end{array} \mathrm{id}^{i\,|\$\Lambda(\boldsymbol{p})|} \begin{array}{c} u_2 \\ \downarrow \\ u_2 \end{array} \quad \text{and} \quad (\$\Lambda(\boldsymbol{p}))^j \begin{array}{c} u_1 \\ \downarrow \\ v_1' \end{array} \begin{array}{c} a \\ \downarrow \\ a \end{array} \mathrm{id}^{j\,|\$\Lambda(\boldsymbol{p})|} \begin{array}{c} u_2 \\ \downarrow \\ u_2 \end{array} \quad \text{for } \mathcal{T}.$$

Thus, $(\$\Lambda(\boldsymbol{p}))^i$ and $(\$\Lambda(\boldsymbol{p}))^j$ must already act differently on u_1, which is from Γ^* and, thus, also a witness for $\mathcal{R}$.

Next, we observe that $*$ is not changed by the action of any state and that we have $q \cdot * = \mathrm{id}$ for all $q \in Q \setminus \{s\}$ and $s \cdot * = t$. Thus, $\mathscr{G}(\mathcal{T}) \cdot *$ is the subgroup T generated by t in $\mathscr{G}(\mathcal{T})$ and we obtain $\mathscr{G}(\mathcal{T}) \cdot w = T \cdot w'$. To understand the elements in $T \cdot w'$, we will show that we have the cross diagram

$$t^{k\,|\$\Lambda(\boldsymbol{p})|} \begin{array}{c} w' \\ \downarrow \\ w' \end{array} (\$\Lambda(\boldsymbol{p}))^k \quad \text{in } \mathcal{T} \tag{$\dagger$}$$

for all $k \in \mathbb{N}$. This shows that $\mathscr{G}(\mathcal{T}) \cdot w = T \cdot w'$ is given by

$$\mathrm{Suf}\,(\$\Lambda(\boldsymbol{p}))^{-\omega} \cup \mathrm{Suf}\left(\overline{\$\Lambda(\boldsymbol{p})}\right)^{-\omega} \text{ in } \mathscr{G}(\mathcal{T}),$$

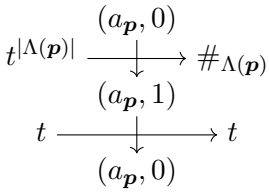
which is finite if and only if $\$\Lambda(\boldsymbol{p})$ has finite order in $\mathscr{G}(\mathcal{T})$ (or, equivalently, in $\mathscr{G}(\mathcal{R})$) by Fact 1.4.2.1.

The easiest way to establish the cross diagrams ($\dagger$) is by calculation. For example, for $\boldsymbol{p} = p_3p_2p_1$, we have $w' = (a_{p_1}, 0)(a_{p_2}, 0)(a_{p_3}, 0)\#$ and the cross diagram:

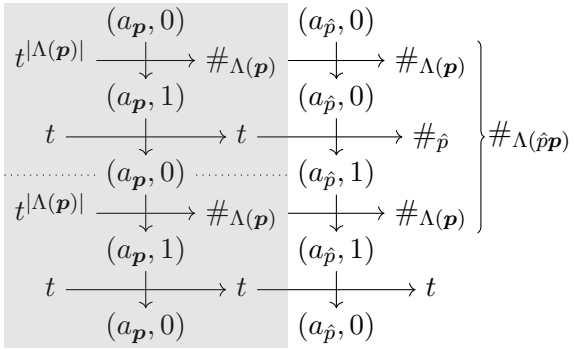
$$\begin{array}{c}
\begin{array}{c} (a_{p_1},0) \\ \downarrow \\ (a_{p_1},0) \end{array}
\begin{array}{c} (a_{p_2},0) \\ \downarrow \\ (a_{p_2},0) \end{array}
\begin{array}{c} (a_{p_3},0) \\ \downarrow \\ (a_{p_3},0) \end{array}
\begin{array}{c} \# \\ \downarrow \\ \# \end{array} \\
t \to \#_{p_1} \to \#_{p_1} \to \#_{p_1} \to p_1 \\
t \to t \to \#_{p_2} \to \#_{p_2} \to p_2 \\
t \to \#_{p_1} \to \#_{p_1} \to \#_{p_1} \to p_1 \\
t \to t \to t \to \#_{p_3} \to p_3 \\
t \to \#_{p_1} \to \#_{p_1} \to \#_{p_1} \to p_1 \\
t \to t \to \#_{p_2} \to \#_{p_2} \to p_2 \\
t \to \#_{p_1} \to \#_{p_1} \to \#_{p_1} \to p_1 \\
t \to t \to t \to t \to \$
\end{array}
\begin{array}{l}
\left.\begin{array}{c} \\ \\ \\ \end{array}\right\} \Lambda(q_2q_1) \\
\\
\left.\begin{array}{c} \\ \\ \\ \end{array}\right\} \Lambda(q_2q_1)
\end{array}
\left.\begin{array}{c} \\ \\ \\ \\ \\ \\ \\ \\ \end{array}\right\} \Lambda(q_3q_2q_1)$$

Notice that, in the second component, t implements a binary increment (in the same way as the adding machine in Example 0.2.1.4). This is what creates the pattern of $\Lambda(\boldsymbol{p})$.

For a formal proof, we first define the shorthand notations $(a_{\boldsymbol{p}}, i) = (a_{p_1}, i)\dots(a_{p_\ell}, i)$ for $i \in \{0,1\}$ and $\boldsymbol{p} = p_\ell \dots p_1$ as well as $\#_\varepsilon = \varepsilon$ and $\#_{\Lambda(\hat{p}\boldsymbol{p})} = \#_{\Lambda(\boldsymbol{p})}\#_{\hat{p}}\#_{\Lambda(\boldsymbol{p})}$ for $\boldsymbol{p} \in P^*$ and $\hat{p} \in P$. We start by showing the cross diagram(s)

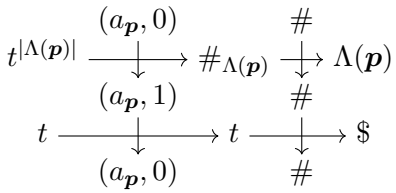


for every $\boldsymbol{p} \in P^+$ by induction on the length of $\boldsymbol{p}$. For $\boldsymbol{p} = p \in P$, this is easily verified from the definition of $\mathcal{T}$ (note that $\Lambda(\boldsymbol{p}) = \Lambda(p) = p$ in this case). For $\boldsymbol{p}' = \hat{p}\boldsymbol{p}$ with $\hat{p} \in P$, we have $|\Lambda(\hat{p}\boldsymbol{p})| = 2|\Lambda(\boldsymbol{p})| + 1$ and the cross diagram



where the shaded part is obtained by using the induction hypothesis twice. For the part on the right, notice that we have $\#_p \circ (a_q, i) = (a_q, i)$ and $\#_p \cdot (a_q, i) = \#_p$ for all $p, q \in P$ and $i \in \{0,1\}$ by construction. The two transactions on the right involving t can directly be verified, which concludes the induction.

Finally, we can extend this to prove the cross diagrams (†) required above:



The only point to notice here is that we indeed have $\#_{\Lambda(\boldsymbol{p})} \cdot \# = \Lambda(\boldsymbol{p})$; however, this is straight-forward to verify. □

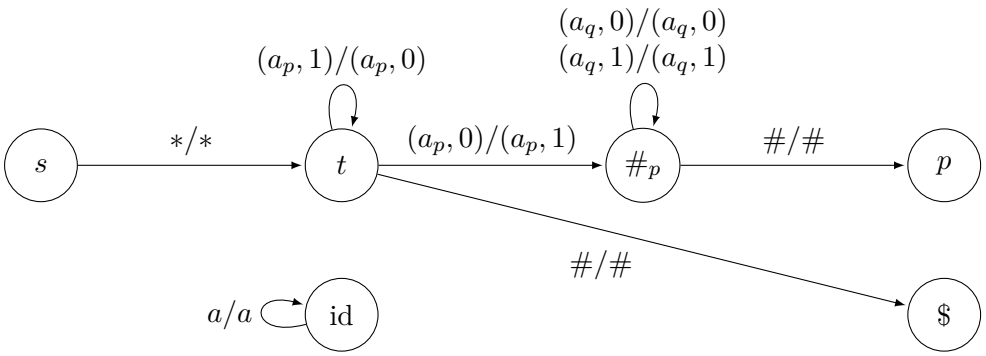


Figure 2.11: New transitions of $\mathcal{T}$

Orbital and Dual Formulation. Using the results from Section 1.4, we can re-formulate Automaton Group Finiteness in two interesting ways. First, we obtain from Corollary 1.4.1.14 (and Fact 0.3.2.2) that it is equivalent to (the complement of) the problem:

Input: a $\mathscr{G}$-automaton $\mathcal{T} = (Q, \Sigma, \delta)$
Question: $\exists \alpha \in \Sigma^\omega : |Q^* \circ \alpha| = \infty$?

This view allows us to also re-formulate Theorem 2.3.3.2.

Corollary 2.3.3.3 ([8, Corollary 3.2])**.** *The decision problem*

Constant: *a $\mathscr{G}$-automaton $\mathcal{T} = (Q, \Sigma, \delta)$*
Input: *a finite word $w \in \Sigma^*$*
Question: $\exists \alpha \in \Sigma^\omega : |Q^* \circ w\alpha| = \infty$*?*

is undecidable for some $\mathscr{G}$-automaton $\mathcal{T}$.

Proof (of [8, Corollary 3.2]). We have to show that $\mathscr{G}(\mathcal{T}) \cdot w$ is infinite if and only if there is some ω-word $\alpha \in \Sigma^\omega$ such that the orbit $Q^* \circ w\alpha$ is infinite.

In $\mathscr{G}(\mathcal{T})$, the elements of $\mathscr{G}(\mathcal{T}) \cdot w$ are given by $\widetilde{Q}^* \cdot w$, which is suffix-closed and, thus, by Theorem 1.4.1.13, infinite in $\mathscr{G}(\mathcal{T})$ if and only if there is some $\alpha \in \Sigma^\omega$ with $|\widetilde{Q}^* \cdot w \circ \alpha| = \infty$. We claim that $\widetilde{Q}^* \cdot w \circ \alpha$ is infinite if and only if $\widetilde{Q}^* \circ w\alpha$ is. Since the latter is the case if and only if $Q^* \circ w\alpha$ is infinite by Fact 1.4.1.15, we are done when we have shown this claim.

Clearly, we can map $\widetilde{Q}^* \circ w\alpha$ surjectively onto $\widetilde{Q}^* \cdot w \circ \alpha$ by removing the prefix of length $|w|$. Thus, if $\widetilde{Q}^* \cdot w \circ \alpha$ is infinite, so must be $\widetilde{Q}^* \circ w\alpha$. On the other hand, we have $\widetilde{Q}^* \circ w\alpha \subseteq \left(\widetilde{Q}^* \circ w\right)\left(\widetilde{Q}^* \cdot w \circ \alpha\right)$ and the first of the two sets on the right is always finite. Thus, if $\widetilde{Q}^* \circ w\alpha$ is infinite, $\widetilde{Q}^* \cdot w \circ \alpha$ must also be infinite. □

Since a $\mathscr{G}$-automaton generates an infinite group if and only if its dual generates an infinite semigroup (which follows, for example, from combining Corollary 1.4.1.12 and Fact 1.4.1.15), we can re-formulate Automaton Group Finiteness in yet another way. It is equivalent to the problem:

Input: a complete and reversible $\mathscr{S}$-automaton $\mathcal{T} = (Q, \Sigma, \delta)$
Question: is $\mathscr{S}(\mathcal{T})$ finite?

Again, we can re-formulate Theorem 2.3.3.2 under this view and obtain that there is a semigroup generated by a complete and reversible automaton whose finiteness problem for left principal ideals is undecidable.

Corollary 2.3.3.4 ([8, Corollary 3.3])**.** *The decision problem*

Constant: *a complete and reversible $\mathscr{S}$-automaton $\mathcal{R} = (P, \Gamma, \tau)$*
Input: *a finite state sequence $\boldsymbol{p} \in P^*$*
Question: *is $P^*\boldsymbol{p}$ finite in $\mathscr{S}(\mathcal{R})$?*

is undecidable.

Proof (adapted from the proof of [8, Corollary 3.3]). We reduce the problem from Corollary 2.3.3.3 to this problem. As the automaton $\mathcal{R}$, we choose the dual $\partial\mathcal{T}$ of the $\mathscr{G}$-automaton $\mathcal{T} = (Q, \Sigma, \delta)$ and, for the reduction, we map $w \in \Sigma^*$ to ∂w as the input sequence $\boldsymbol{p}$. We have to show that there is some $\alpha \in \Sigma^\omega$ with $|Q^* \circ w\alpha| = \infty$ if and only if $\Sigma^*\partial w$ is infinite in $\mathscr{S}(\partial\mathcal{T})$.

If the orbital graph $Q^* \circ w\alpha$ is infinite (for some $\alpha \in \Sigma^\omega$), it must contain an infinite simple path labeled by some $\pi \in Q^{-\omega}$ starting in $w\alpha$. For this π, we have $|\operatorname{Suf} \pi \circ w\alpha| = \infty$ and, by Corollary 1.4.1.19, also that $\operatorname{Suf} \partial(w\alpha) \circ_\partial \partial\pi = \operatorname{Suf}((\partial\alpha)(\partial w)) \circ_\partial \partial\pi \subseteq \Sigma^*\partial w \circ_\partial \partial\pi$ is infinite. This is only possible if $\Sigma^*\partial w$ is infinite in $\mathscr{S}(\partial\mathcal{T})$.

On the other hand, if $\Sigma^*\partial w$ is infinite in $\mathscr{S}(\partial\mathcal{T})$, we have in particular that $L = \Sigma^*\partial w \cup \operatorname{Suf} \partial w$ is infinite in $\mathscr{S}(\partial\mathcal{T})$. Since L is suffix-closed, we obtain by Theorem 1.4.1.13 that there is some $\pi \in Q^{-\omega}$ such that $L \circ_\partial \partial\pi$ is infinite. This is only possible if $\Sigma^*\partial w \circ_\partial \partial\pi$ is infinite. This means that there is some $\alpha \in \Sigma^\omega$ such that $(\partial\alpha)(\partial w)$ labels a path in the orbital graph $\Sigma^* \circ_\partial \partial\pi$ starting in $\partial\pi$ which visits infinitely many nodes, i. e. we have that $\operatorname{Suf}((\partial\alpha)(\partial w)) \circ_\partial \partial\pi = \operatorname{Suf} \partial(w\alpha) \circ_\partial \partial\pi$ is infinite. By Corollary 1.4.1.19, this means that $\operatorname{Suf} \pi \circ w\alpha \subseteq Q^* \circ w\alpha$ must be infinite as well. □

2.4 Expandability

This section is devoted to the study of the notion of "expandability". Informally, a word w is expandable if we can append some suffix to w and obtain a larger orbit. From Corollary 1.4.1.14, we know that every infinite automaton semigroup admits a word with an infinite orbit. However, the proof for this is purely existential and we do not gain much information about the structure of the word. Gaining such information is the idea behind the notion of expandability. Clearly, if α is an ω-word with an infinite orbit, then every finite prefix of α is expandable. On the other hand, if we start with an expandable word and successively append orbit increasing suffixes in such a way that the resulting word always remains expandable, then, in the limit, we obtain an ω-word with an infinite orbit. Therefore, studying expandable words is closely related to studying the words with infinite orbital or – in the group case – Schreier graphs.

Formally, we define expandability in the following way.

Definition 2.4.0.1. Let $\mathcal{T} = (Q, \Sigma, \delta)$ be an $\mathscr{S}$-automaton, $K \subseteq Q^*$ and $k \in \mathbb{Z}$.

A finite word $w \in \Sigma^*$ is *K-orbit k-expandable* (with respect to $\mathcal{T}$) if there is some $x \in \Sigma^*$ such that $|K \circ wx| \geq |K \circ w| + k$. A finite word $w \in \Sigma^*$ is *K-orbit expandable* if it is K-orbit k-expandable for some $k \geq 1$.

An important observation here is that, if the $\mathscr{S}$-automaton $\mathcal{T}$ is partial, the size of $K \circ wx$ can indeed be smaller than the size of $K \circ w$, while in the complete case this cannot happen.

We are primarily interested in the cases $K = Q^*$ and $K = \boldsymbol{P}^*$ for a finite set $\boldsymbol{P} \subseteq Q^*$, which belong to the orbit of the whole semigroup (or group) and the orbit of a finitely generated subsemigroup (or subgroup). To make this simpler, we say that a finite word $w \in \Sigma^*$ is *k-expandable* (*expandable*) if it is Q^*-orbit k-expandable (Q^*-orbit expandable).

In the first subsection, we will see that it is decidable whether a given word is $\boldsymbol{P}^*$-orbit k-expandable with respect to some given $\mathscr{S}$-automaton for some given finite set $\boldsymbol{P}$ of state sequences, i. e. whether the $\boldsymbol{P}^*$-orbit can be increased by appending a suffix. This contrasts the result from Corollary 2.3.3.3 where the question was whether an infinite suffix can be appended to obtain an infinite orbit. From the decision algorithm, we obtain an upper bound on the length of a shortest suffix that increases the $\boldsymbol{P}^*$-orbit by k.

Afterwards, we look at the special case where the automaton is a $\mathscr{G}$-automaton. Here, we give a (somewhat) algebraic characterization for the expandable words using their so-called shifted stabilizer. We use this characterization to obtain a more efficient algorithm in the group case to decide whether a given word is expandable and to obtain a better upper bound on the length of a shortest orbit increasing suffix. Again, we prove these results directly for finitely generated subgroups.

Then, we apply our results to the class of automaton groups of bounded activity. We will discuss the importance of this class later in Subsection 2.4.3 but point out that many decision problems are in fact easier for such automata than in the general case (for example, the word problem discussed in Section 2.1). Using our results about the expandability in groups, we will construct a (finite) weighted acceptor that describes the $\boldsymbol{P}^*$-orbit size of a given word where $\boldsymbol{P}$ is a finite set of state sequences. This description yields some interesting consequences. Most notably, we obtain that the finiteness problem and the problem whether the group acts spherically transitive are both decidable for automaton groups of bounded activity.[84] We even obtain these two results for finitely generated subgroups of automaton groups of bounded activity.[85]

[84] Both problems are open in the case of general automaton groups (see Section 2.3 and [GNS00, 7.2 e) and f)]).

[85] ... answering [2, Question 3.7]

Attribution. The notion of expandability was developed together with Daniele D'Angeli and Emanuele Rodaro [4]. Here, we generalize it to K-orbits where K is a language over the state set. The decidability results and the upper bounds in the general case and for groups that we present here are direct generalizations of the corresponding results in [4] to finitely generated subsemigroups and subgroups and we (mostly) follow the presentation given in [4]. In fact, some proofs are verbatim copies from [4] adapted to our generalized setting.

The activity hierarchies (and, thus, bounded automaton groups) were introduced by Sidki [Sid00]. The results given here for automaton groups of bounded activity have not previously been published in this form. However, the author of the current work approached Ievgen Bondarenko with (a preliminary version not handling subgroups) of the results, who turned out to already have had solved the finiteness problem for automaton groups of bounded activity some time ago but never published the result.

This resulted in a joint paper on this topic [2], whose approach deviates from the one given here, however. In fact, the approach presented here is already generalized to finitely generated subgroups and, thus, solves [2, Question 3.7].

2.4.1 Expandability is Decidable

In this subsection, we show that it is decidable whether a given word is $\boldsymbol{P}^*$-orbit expandable for a (possibly) given finite set $\boldsymbol{P}$ of state sequences with respect to a (possibly) given $\mathscr{S}$-automaton. In fact, we give a nondeterministic space-bounded algorithm and analyze its resource requirements.

The key idea of our algorithm is to basically guess a suffix x such that the $\boldsymbol{P}^*$-orbit of wx is larger than that of the input word w. To verify that the $\boldsymbol{P}^*$-orbit of wx is indeed larger, we will guess many different simple paths in a generalized orbital graph of wx. To bound the length of these paths, we will use the following combinatorial lemma, which holds for arbitrary languages K over the state set.

Lemma 2.4.1.1 (extension of [4, Lemma 3.1] to K-orbits)**.** *Let $\mathcal{T} = (Q, \Sigma, \delta)$ be an $\mathscr{S}$-automaton and let $w \in \Sigma^*$ be K-orbit k-expandable for some $K \subseteq Q^*$. Then, there is an $x \in \Sigma^*$ such that*

$$n + k \leq |K \circ wx| < (n + k)\,|\Sigma|,$$

where $n = |K \circ w|$.

Proof (of [4, Lemma 3.1] adapted to K-orbits). Since w is K-orbit k-expandable, there is a $y \in \Sigma^*$ with $n + k \leq |K \circ wy|$. Let x' denote the longest prefix of y such that $|K \circ wx'| < n + k$, i. e. we have $y = x'ay'$ for some $a \in \Sigma$ and some $y' \in \Sigma^*$. By our choice of x', we have $n + k \leq |K \circ wx'a|$. As the size of $|K \circ ua|$ is limited by $|K \circ u|\,|\Sigma|$ for any word $u \in \Sigma^*$, this yields

$$n + k \leq \left|K \circ wx'a\right| \leq \left|K \circ wx'\right|\,|\Sigma| < (n + k)\,|\Sigma|\,.$$

Thus, $x = x'a$ satisfies the inequality in the lemma. □

$\boldsymbol{P}^*$-Orbital Graphs. For the algorithm, we are mostly interested in the case where K is of the form $K = \boldsymbol{P}^*$ for a finite set $\boldsymbol{P}$ of state sequences (because this belongs to the orbit of the subsemigroup generated by $\boldsymbol{P}$ in the semigroup). To make this easier, we enrich the $\boldsymbol{P}^*$-orbit with a graph structure (similar to the normal orbital graph). For an $\mathscr{S}$-automaton $\mathcal{T} = (Q, \Sigma, \delta)$ and a finite set $\boldsymbol{P} \subseteq Q^*$, the *$\boldsymbol{P}^*$-orbital graph* of a word $w \in \Sigma^\infty$ consists of the nodes $\boldsymbol{P}^* \circ w$ and its edges are given by

$$\{\boldsymbol{p} \circ u \xleftarrow{\boldsymbol{p}} u \mid \boldsymbol{p} \in \boldsymbol{P}, u \in \boldsymbol{P}^* \circ w, \boldsymbol{p} \circ u \text{ defined}\}.$$

In the same way as with $Q^* \circ w$, we do not distinguish between $\boldsymbol{P}^* \circ w$ as a set and as a graph. However, it is important to point out that the edges are labeled with state sequences in $\boldsymbol{P}^* \circ w$ in general and not only with single states. Therefore, $\boldsymbol{P}^* \circ w$, may not be a subgraph of $Q^* \circ w$.

Deciding Expandability. In the algorithm, we will need to compute the size of the $\boldsymbol{P}^*$-orbit of the input word. That this can basically be done in linear nondeterministic space is stated in the following fact.

Fact 2.4.1.2. *The problem*

Input: *an $\mathscr{S}$-automaton $\mathcal{T} = (Q, \Sigma, \delta)$,*
a finite set $\boldsymbol{P} \subseteq Q^*$,
a natural number n and a word $w \in \Sigma^*$
Question: *is $|\boldsymbol{P}^* \circ w| = n$?*

is in $\text{NSpace}(|w|(1 + \log|\Sigma|) + \log|\boldsymbol{P}|)$.

Proof. By simply guessing n words from $\Sigma^{|w|}$ one after another (using a suitable linear ordering of Σ^*) and checking that each guessed word can be reached from w in the graph $\boldsymbol{P}^* \circ w$, one sees easily that the modified problem with question "is $|\boldsymbol{P}^* \circ w| \geq n$?" is in $\text{NSpace}(|w|(1 + \log|\Sigma|) + \log|\boldsymbol{P}|)$. Note that n must not exceed $|\Sigma|^{|w|}$ (or the answer will be "no" anyway), so we can store a counter for a value up to n in the given space. Since nondeterministic space classes are closed under complement (see, e. g., [Pap94, Theorem 7.6]), we obtain the fact. □

We can now describe and analyze the algorithm in more detail. We obtain that the problem is in ExpSpace.

Theorem 2.4.1.3 (extension of [4, Theorem 3.2] to $\boldsymbol{P}^*$-orbits)**.** *The problem*

Input: *an $\mathscr{S}$-automaton $\mathcal{T} = (Q, \Sigma, \delta)$,*
a finite set $\boldsymbol{P} \subseteq Q^*$,
a natural number k and a word $w \in \Sigma^*$
Question: *is w $\boldsymbol{P}^*$-orbit k-expandable (with respect to $\mathcal{T}$)?*

is in $\text{NSpace}((n+k)^2\,|\Sigma|\,(1 + \log|\boldsymbol{P}|) + |w|(1 + \log|\Sigma|)) \subseteq \text{ExpSpace}$, *where* $n = |\boldsymbol{P}^* \circ w| \leq |\Sigma|^{|w|}$ *is the size of the $\boldsymbol{P}^*$-orbit of the input word. This yields that the problem*

Constant: *an $\mathscr{S}$-automaton $\mathcal{T} = (Q, \Sigma, \delta)$ and*
a finite set $\boldsymbol{P} \subseteq Q^*$
Input: *a natural number k and a word $w \in \Sigma^*$*
Question: *is w $\boldsymbol{P}^*$-orbit k-expandable (with respect to $\mathcal{T}$)?*

is in $\text{NSpace}\left(2^{\mathcal{O}(|w|+\log k)}\right) \subseteq \text{ExpSpace}$.

Proof (based on the proof of [4, Theorem 3.2]). First, note that all words in $Q^* \circ w$ are of length $|w|$. Thus, n is bounded by $|\Sigma|^{|w|}$. Using Fact 2.4.1.2, we can compute the exact value of n within the required space bound by increasing a variable in a loop starting at 1 and going to $|\Sigma|^{|w|}$ until the value is found.

We can solve the main part of the problem using a "guess and check" approach. We give a rather informal description of the algorithm here; pseudo-code for it can be found in Algorithm 2. First, we guess $n+k$ state sequences $\boldsymbol{p}_1, \ldots, \boldsymbol{p}_{n+k} \in \boldsymbol{P}^*$ of length smaller than $N = (n+k)|\Sigma| - 1$. The idea is that these state sequences lead to different elements

in $\boldsymbol{P}^* \circ wx$ for some witness $x \in \Sigma^*$ for the $\boldsymbol{P}^*$-orbit k-expandability of w. Next, we compute $\boldsymbol{p}_i \cdot w$ for each of the state sequences. Finally, we guess x letter by letter (without storing the previous letters). After we guessed a new letter $b \in \Sigma$, we update the stored state sequences $\boldsymbol{p}_i$ to $\boldsymbol{p}_i \cdot b$. While we do all of this, we also keep track of the pairs of state sequences for which we have already encountered a difference in their outputs on w followed by the guessed letters. Whenever a transition is not defined, we simply cancel the respective computational branch.

It is clear that, if the algorithm returns "w is $\boldsymbol{P}^*$-orbit k-expandable", then this is correct as the guessed state sequences and letters witness the expandability. On the other hand, it is sufficient to only consider state sequences whose length over $\boldsymbol{P}$ is smaller than N to discover a witness if one exists. To see this, suppose that w is $\boldsymbol{P}^*$-orbit k-expandable. Then, by Lemma 2.4.1.1, there is an $x \in \Sigma^*$ such that w is $\boldsymbol{P}^*$-orbit k-expandable by x and the $\boldsymbol{P}^*$-orbital graph is of size $|\boldsymbol{P}^* \circ wx| \leq N$. Now, any element of this graph is reachable from wx by a path with less than N edges as longer paths need to include a loop by the pigeon hole principle. The labels of these paths correspond to the elements $\boldsymbol{p}_i$ and, thus, bounding them to elements from $\boldsymbol{P}^{<N}$ is no restriction.

The most interesting part of the space analysis are the variables $\boldsymbol{p}_1, \ldots, \boldsymbol{p}_{n+k}$ and the variable `differences`. Other variables, like elements of Σ and counters up to $|w|$, n, $n+k$ or N, can certainly be realized in $\mathcal{O}(1 + \log|\Sigma| + \log|w| + \log(n+k))$ and, thus, in the space bound stated in the theorem. For storing a single variable $\boldsymbol{p}_i \in \boldsymbol{P}^{<N}$, we need space smaller than $N(1+\log|\boldsymbol{P}|) < (n+k)\,|\Sigma|\,(1+\log|\boldsymbol{P}|)$. Thus, for all variables $\boldsymbol{p}_1, \ldots, \boldsymbol{p}_{n+k}$, we need less than $(n+k)^2\,|\Sigma|\,(1+\log|\boldsymbol{P}|)$ space, which is within the required space bound. Finally, for `differences`, we need to store a bit for the $\binom{n+k}{2}$ many subsets of size 2 of $\{1, \ldots, n+k\}$. This yields a space requirement in $\mathcal{O}\left(\binom{n+k}{2}\right) \subseteq \mathcal{O}((n+k)^2)$. □

By counting the possible pairwise distinct configurations on an accepting computational branch of Algorithm 2, we can obtain an upper bound on the length of a shortest word x witnessing the $\boldsymbol{P}^*$-orbit k-expandability of the input word w.

Corollary 2.4.1.4 (extension of [4, Corollary 3.3] to K-orbits)**.** *Let $\mathcal{T} = (Q, \Sigma, \delta)$ be an $\mathscr{S}$-automaton and let $\boldsymbol{P} \subseteq Q^*$ be finite. A word $w \in \Sigma^*$ is $\boldsymbol{P}^*$-orbit k-expandable with respect to $\mathcal{T}$ if and only it is already $\boldsymbol{P}^*$-orbit k-expandable by some $x \in \Sigma^*$ with*

$$|x| < (\max\{2, |\boldsymbol{P}|\})^{|\Sigma|\,(n+k)^2\,2^{\binom{n+k}{2}}},$$

where $n = |\boldsymbol{P}^* \circ w|$.

Proof (extension of the proof of [4, Corollary 3.3] to $\boldsymbol{P}^*$-orbits). If w is k-expandable, then Algorithm 2 will return "w is $\boldsymbol{P}^*$-orbit k-expandable" on some computational branch. The letters guessed at Line 32 for this branch yield a witness $x \in \Sigma^*$ for which $|\boldsymbol{P}^* \circ wx| \geq n+k$ holds. However, if, at any two points of the branch, the variables $\boldsymbol{p}_1, \ldots, \boldsymbol{p}_{n+k}$ and `differences` have the same values, then the computation has a cycle and we can shorten x by that cycle. This means that, without loss of generality, we may assume the length of x to be bounded by the number of different configurations of these variables:

$$|x| \leq \left|\boldsymbol{P}^{<N}\right|^{n+k} \cdot 2^{\binom{n+k}{2}}.$$

Algorithm 2 A nondeterministic algorithm to decide whether a word is $\boldsymbol{P}^*$-orbit k-expandable (extension of [4, Algorithm 1] to $\boldsymbol{P}^*$-orbits)

```
1  function IskExpandable(𝒯 = (Q, Σ, δ), P ⊆ Q*, k, w = a_1 ... a_m): 𝔹;
2  const
3    n = |P* ∘ w|;
4    N = (n + k)|Σ| − 1;
5  var
6    p_1, ..., p_{n+k} ∈ P^{<N};
7    p ∈ P;
8    differences ⊆ {{i, j} | i ≠ j, 1 ≤ i, j ≤ n + k};        ▷ For which pairs have we seen a difference?
9    b ∈ Σ;
10 begin
11   for i ∈ {1, ..., n + k} do        ▷ Guess initial values p_1^(0), ..., p_{n+k}^(0) ∈ P^{<N} of p_1, ..., p_{n+k}
12     p_i ← ε;
13     while guess({true, false}) and |p_i| < N − 1 do
14       p ← guess(P);                 ▷ We store the elements of P as indices (using space log |P|)
15       p_i ← p_i p;
16     od;
17   od;
18   differences ← ∅;                  ▷ Compute p_i^(0) · w while checking for differences between pairs
19   for ℓ ∈ {1, ..., m} do
20     if ∃i : p_i ∘ a_ℓ is undefined then
21       fail;
22     fi;
23     differences ← differences ∪ {{i, j} | p_i ∘ a_ℓ ≠ p_j ∘ a_ℓ};
24     for i ∈ {1, ..., n + k} do
25       p_i ← p_i · a_ℓ;
26     od;
27   od;
28   while true do                     ▷ Guess x ∈ Σ* letter-wise until we have seen a difference for every pair
29     if ∀1 ≤ i, j ≤ n + k : {i, j} ∈ differences then
30       return "w is K-orbit k-expandable";   ▷ All p_i^(0) ∘ wx are defined and pairwise disjoint
31     fi;
32     b ← guess(Σ);                                          ▷ Guess next letter of x
33     if ∃i : p_i ∘ b is undefined then
34       fail;
35     fi;
36     differences ← differences ∪ {{i, j} | p_i ∘ b ≠ p_j ∘ b};
37     for i ∈ {1, ..., n + k} do
38       p_i ← p_i · b;
39     od;
40   od;
41 end;
```

For $|\boldsymbol{P}| \geq 2$, we have

$$\left|\boldsymbol{P}^{<N}\right| = \sum_{i=0}^{N-1} |\boldsymbol{P}|^i = \frac{|\boldsymbol{P}|^N - 1}{|\boldsymbol{P}| - 1} < |\boldsymbol{P}|^N < |\boldsymbol{P}|^{|\Sigma|\,(n+k)}$$

and, for $|\boldsymbol{P}| = 1$, we have

$$\left|\boldsymbol{P}^{<N}\right| = \sum_{i=0}^{N-1} |\boldsymbol{P}|^i = N < 2^N < 2^{|\Sigma|\,(n+k)}. \qquad \square$$

2.4.2 Expandability in Groups

In the group case, we can state an algebraic characterization of expandable words based on their stabilizers and this characterization yields a more efficient algorithm for deciding whether a word is expandable.

If $\mathcal{T} = (Q, \Sigma, \delta)$ is a $\mathscr{G}$-automaton and $\boldsymbol{P} \subseteq \widetilde{Q}^*$ is a finite set, then we can see easily from Fact 0.3.2.2 that a word $w \in \Sigma^*$ is $\boldsymbol{P}^*$-orbit expandable (with respect to $\widetilde{\mathcal{T}}$) if and only if it is $\widetilde{\boldsymbol{P}}^*$-orbit expandable. Therefore, we will not make a distinction between the two. To lighten the notation further, we simply write $\mathrm{Stab}_{\boldsymbol{P}}(w)$ instead of $\widetilde{\boldsymbol{P}}^* \cap \mathrm{Stab}^{\mathbb{1}}_{\widetilde{\mathcal{T}}}(w)$ for $w \in \Sigma^*$ whenever the $\mathscr{G}$-automaton $\mathcal{T}$ is clear from the context.

With this notation, we can show a connection between the $\boldsymbol{P}^*$-orbit size of wx on the one hand and, on the other hand, the $\boldsymbol{P}^*$-orbit size of w and the state sequences from

$$\mathrm{Stab}_{\boldsymbol{P}}(w) \cdot w = \{\boldsymbol{p} \cdot w \mid \boldsymbol{p} \in \mathrm{Stab}_{\boldsymbol{P}}(w)\}.$$

This set $\mathrm{Stab}_{\boldsymbol{P}}(w) \cdot w$ contains the state sequences from $\widetilde{\boldsymbol{P}}^*$ stabilizing w *shifted* by w. It is easy to see that it is closed under product and taking inverses and, thus, forms a subgroup in $\mathscr{G}(\mathcal{T})$, which we call the *shifted $\boldsymbol{P}$-stabilizer* of w. The connection between the sifted stabilizer and the two mentioned orbit sizes is given in the next lemma.

Lemma 2.4.2.1 (special case of [4, Lemma 4.1] extended to subgroups)**.** *Let $\mathcal{T} = (Q, \Sigma, \delta)$ be a $\mathscr{G}$-automaton and let $\boldsymbol{P} \subseteq \widetilde{Q}^*$ be finite. For $w, x \in \Sigma^*$, we have*

$$|\boldsymbol{P}^* \circ wx| = |\boldsymbol{P}^* \circ w| \cdot |\mathrm{Stab}_{\boldsymbol{P}}(w) \cdot w \circ x|\,.$$

Proof. First note that we have $\boldsymbol{P}^* \circ wx = \widetilde{\boldsymbol{P}}^* \circ wx$ and $\boldsymbol{P}^* \circ w = \widetilde{\boldsymbol{P}}^* \circ w$ by Fact 0.3.2.2. Therefore, we will not distinguish between the two.

Let G be the subgroup given by $\widetilde{\boldsymbol{P}}^*$ in $\mathscr{G}(\mathcal{T})$, H be the subgroup formed by $\mathrm{Stab}_{\boldsymbol{P}}(w)$ in $\mathscr{G}(\mathcal{T})$ and K be the subgroup formed by $\mathrm{Stab}_{\boldsymbol{P}}(wx)$ in $\mathscr{G}(\mathcal{T})$. It is easy to see that we have $K \leq_{\mathscr{G}} H \leq_{\mathscr{G}} G \leq_{\mathscr{G}} \mathscr{G}(\mathcal{T})$ (where we use $\leq_{\mathscr{G}}$ to indicate a subgroup relation). We define an action of H on Σ^* by defining $h \star y$ for $h \in H$ and $y \in \Sigma^*$ as $\boldsymbol{p} \cdot w \circ y$ where $\boldsymbol{p} \in \widetilde{\boldsymbol{P}}^*$ is in $\mathscr{G}(\mathcal{T})$ equal to h. This action is well-defined by Fact 0.3.1.1. Let

$$\mathrm{Stab}_\star(x) = \{\boldsymbol{s} \in \mathrm{Stab}_{\boldsymbol{P}}(w) \mid \boldsymbol{s} \cdot w \circ x = x\} \subseteq \mathrm{Stab}_{\boldsymbol{P}}(w)$$

and observe that we have $\mathrm{Stab}_{\boldsymbol{P}}(wx) = \mathrm{Stab}_\star(x)$. Therefore, the image of $\mathrm{Stab}_\star(x)$ in $\mathscr{G}(\mathcal{T})$ (and, thus, in G and H) is K. We obtain

$$|\,\mathrm{Stab}_{\boldsymbol{P}}(w) \cdot w \circ x| = |H \star x| = |H/K|$$

where the equality on the left follows from the set equality $\mathrm{Stab}_{\boldsymbol{P}}(u) \cdot u \circ x = H \star x = \{h \star x \mid h \in H\}$ and the equality on the right is a well-known fact of group orbits. Therefore, we finally have

$$|\boldsymbol{P}^* \circ wx| = |G/K| = |G/H| \cdot |H/K| = |\boldsymbol{P}^* \circ w| \cdot |\,\mathrm{Stab}_{\boldsymbol{P}}(w) \cdot w \circ x|. \qquad \square$$

We immediately obtain the following proposition from Lemma 2.4.2.1.

Proposition 2.4.2.2 (extension of [4, Proposition 4.2] to subgroups)**.** *Let $\mathcal{T} = (Q, \Sigma, \delta)$ be a $\mathscr{G}$-automaton and let $\boldsymbol{P} \subseteq \widetilde{Q}^*$ be finite. A word $w \in \Sigma^*$ is $\boldsymbol{P}^*$-orbit expandable if and only if*

$$\mathrm{Stab}_{\boldsymbol{P}}(w) \cdot w \neq \{\mathbb{1}\} \text{ in } \mathscr{G}(\mathcal{T}).$$

This characterization leads to a more efficient algorithm to decide the $\boldsymbol{P}^*$-orbit expandability of a given word w in the group case. The idea is to nondeterministically guess an element of $\mathrm{Stab}_{\boldsymbol{P}}(w) \cdot w$ which is non-trivial in the automaton group. However, to get to an actual decision algorithm, we need to bound the length of the guessed element somehow and this is what we use the following lemma for. It states that the shifted stabilizer of w is generated by a set of "short" generators.

Lemma 2.4.2.3 (extension of [4, Lemma 4.4] to subgroups)**.** *Let $\mathcal{T} = (Q, \Sigma, \delta)$ be a $\mathscr{G}$-automaton, let $\boldsymbol{P} \subseteq \widetilde{Q}^*$ be finite and let $w \in \Sigma^*$. Then, there is some $C \subseteq \widetilde{\boldsymbol{P}}^{<2n}$ with*

$$\left(\widetilde{C \cdot w}\right)^* = (\widetilde{C} \cdot w)^* = \mathrm{Stab}_{\boldsymbol{P}}(w) \cdot w \text{ in } \mathscr{G}(\mathcal{T})$$

where $n = |\boldsymbol{P}^* \circ w|$.

Proof (compare to proof of [4, Lemma 4.4]). First, we will show that there is some $C \subseteq \widetilde{\boldsymbol{P}}^{<2n}$ with $\widetilde{C}^* = \mathrm{Stab}_{\boldsymbol{P}}(w)$ in $\mathscr{G}(\mathcal{T})$. Clearly, the elements of $\mathrm{Stab}_{\boldsymbol{P}}(w)$ correspond to the loops starting and ending in w in the $\widetilde{\boldsymbol{P}}^*$-orbital graph $\widetilde{\boldsymbol{P}}^* \circ w$. Note that, by Fact 0.3.2.2, the size of this graph is $|\widetilde{\boldsymbol{P}}^* \circ w| = |\boldsymbol{P}^* \circ w| = n$. Let us consider the graph $\widetilde{\boldsymbol{P}}^* \circ w$ as an undirected graph. If we fix some spanning tree for it, then every edge $v \xleftarrow{\boldsymbol{p}} u$ *not* belonging to this spanning tree induces a loop $\boldsymbol{c} \in \widetilde{\boldsymbol{P}}^*$ beginning and ending in w: first, follow the unique path from w to u on the spanning tree, then, take the edge $v \xleftarrow{\boldsymbol{p}} u$ itself, finally, return from v to w on the spanning tree again. Notice that this loop contains at most $n - 1 + 1 + n - 1 = 2n - 1$ edges since any (reduced) path on the spanning tree can visit any node at most once. There are only finitely many such loops and they generate the set of all loops beginning and ending in w. Let C be the set of the labels (from $\widetilde{\boldsymbol{P}}^*$) belonging to these generating loops. Then, C generates the stabilizer $\mathrm{Stab}_{\boldsymbol{P}}(w)$ in $\mathscr{G}(\mathcal{T})$ (as a subgroup).

To see that $C \cdot w = \{\boldsymbol{c} \cdot w \mid \boldsymbol{c} \in C\}$ generates $\mathrm{Stab}_{\boldsymbol{P}}(w) \cdot w$ in $\mathscr{G}(\mathcal{T})$ (as a subgroup), consider some element $\boldsymbol{d} \in \mathrm{Stab}_{\boldsymbol{P}}(w) \cdot w$. There must be some $\boldsymbol{c} \in \mathrm{Stab}_{\boldsymbol{P}}(w)$ with $\boldsymbol{d} = \boldsymbol{c} \cdot w$ and we can write $\boldsymbol{c}$ as a product of elements from $\widetilde{C}$: $\boldsymbol{c} = \boldsymbol{c}_\ell \cdots \boldsymbol{c}_1$. Since all $\boldsymbol{c}_1, \ldots, \boldsymbol{c}_\ell \in \widetilde{C}$ are, in particular, elements of $\mathrm{Stab}_{\boldsymbol{P}}(w)$ (and, thus, stabilize w), we have

$$\boldsymbol{d} = \boldsymbol{c} \cdot w = (\boldsymbol{c}_\ell \cdots \boldsymbol{c}_1) \cdot w = (\boldsymbol{c}_\ell \cdot w) \ldots (\boldsymbol{c}_1 \cdot w) \in (\widetilde{C} \cdot w)^*.$$

Finally, we observe that we have $\overline{\boldsymbol{s} \cdot w} = \overline{\boldsymbol{s}} \cdot w$ for all state sequences $\boldsymbol{s} \in \widetilde{Q}^*$ that stabilize w (i. e. with $\boldsymbol{s} \circ w = w$). Thus, we have $\widetilde{C \cdot w} = \widetilde{C} \cdot w$. □

Since a subgroup is non-trivial if and only if one of its generators is non-trivial, we can improve the upper bounds from Theorem 2.4.1.3 in the group case.

Theorem 2.4.2.4 (extension of [4, Theorem 4.5] to subgroups)**.** *The problem*

Input: *a $\mathscr{G}$-automaton $\mathcal{T} = (Q, \Sigma, \delta)$,*
a finite set $\boldsymbol{P} \subseteq \widetilde{Q}^*$ and
a word $w \in \Sigma^*$

Question: *is w $\boldsymbol{P}^*$-orbit expandable (with respect to $\widetilde{\mathcal{T}}$)?*

is in $\mathrm{NSPACE}(n(1 + \log|\boldsymbol{P}|) + |w|(1 + \log|\Sigma|))$ *where* $n = |\boldsymbol{P}^* \circ w| \leq |\Sigma|^{|w|}$*. This yields* $\mathrm{NSPACE}(2^{\mathcal{O}(|w|)})$ *for the problem*

Constant: *a $\mathscr{G}$-automaton $\mathcal{T} = (Q, \Sigma, \delta)$ and*
a finite set $\boldsymbol{P} \subseteq \widetilde{Q}^*$

Input: *a word $w \in \Sigma^*$*

Question: *is w $\boldsymbol{P}^*$-orbit expandable (with respect to $\widetilde{\mathcal{T}}$)?*

Proof (compare to the proof of [4, Theorem 4.5]). As in the proof of Theorem 2.4.1.3, we use Fact 2.4.1.2 to compute n within our space bound. Then, we guess a state sequence $\boldsymbol{p} \in \widetilde{\boldsymbol{P}}^{<2n}$. To store such a sequence, we need space $\mathcal{O}(n(1 + \log|\boldsymbol{P}|))$.

The rest of the algorithm is similar to the "guess and check" algorithm for the word problem (Algorithm 1). We compute $\boldsymbol{p} \cdot w$ letter by letter (of w). Simultaneously, we check whether $\boldsymbol{p} \in \mathrm{Stab}_{\boldsymbol{P}}(w)$ by computing $\boldsymbol{p} \circ w$ and comparing the result with w (again, we do this letter by letter). For this, we need to store only single letters ($\mathcal{O}(1 + \log|\Sigma|)$) and some pointer ($\mathcal{O}(\log|w|)$).

Finally, we solve the word problem "$\boldsymbol{p} \cdot w \neq \mathbb{1}$ in $\mathscr{G}(\mathcal{T})$?" by guessing a witness $x \in \Sigma^*$ with $\boldsymbol{p} \cdot w \circ x \neq x$. As before, we guess x letter by letter and update the stored state sequence accordingly. At the same time, we check whether at least one output letter differs from its input letter.

Clearly, if we can guess a state sequence $\boldsymbol{p}$ and a witness $x \in \Sigma^*$ with $\boldsymbol{p} \circ w = w$ but $\boldsymbol{p} \cdot w \circ x \neq x$, the shifted $\boldsymbol{P}$-stabilizer of w is non-trivial and w is $\boldsymbol{P}^*$-orbit expandable (by Proposition 2.4.2.2). If, on the other hand, w is $\boldsymbol{P}^*$-orbit expandable, then its shifted $\boldsymbol{P}$-stabilizer is non-trivial (again, by Proposition 2.4.2.2) and, by Lemma 2.4.2.3, it contains a non-trivial element which is in $\widetilde{\boldsymbol{P}}^{<2n} \cdot w$. Some computational branch of the above algorithm will guess the corresponding element $\boldsymbol{p} \in \widetilde{\boldsymbol{P}}^{<2n}$ and a corresponding witness x for the non-triviality of $\boldsymbol{p} \cdot w$. □

In the same way as with Algorithm 2 (for the general case), we can analyze the last part of the presented algorithm to obtain a better upper bound for the witness x in the group case (compared to the general case presented in Corollary 2.4.1.4).

Corollary 2.4.2.5 (extension of [4, Corollary 4.6] to subgroups)**.** *Let $\mathcal{T} = (Q, \Sigma, \delta)$ be a $\mathscr{G}$-automaton and let $\boldsymbol{P} \subseteq \widetilde{Q}^*$ be a finite set. A word $w \in \Sigma^*$ is $\boldsymbol{P}^*$-orbit expandable if and only if it is already $\boldsymbol{P}^*$-orbit expandable by some $x \in \Sigma^*$ with*

$$|x| < (2\,|\boldsymbol{P}|)^{2n}$$

where $n = |\boldsymbol{P}^* \circ w|$.

Proof (extension of the proof of [4, Corollary 4.6] to subgroups). We use the same analysis as in the proof of Corollary 2.4.1.4: consider the last part of the algorithm, in which the witness x is guessed letter by letter. If, during this guessing, the stored state sequence $(\boldsymbol{p} \cdot w) \cdot x$ has the same value as some time before on the same computational branch, then this computational loop can be eliminated. Thus, we only need to count the possible values for the stored state sequence:

$$\left|\widetilde{\boldsymbol{P}}^{<2n}\right| < |\widetilde{\boldsymbol{P}}|^{2n} = (2\,|\boldsymbol{P}|)^{2n} \qquad \square$$

2.4.3 Groups of Bounded Activity

In this subsection, we have a quick look at the special case of automaton groups of bounded activity. We apply the results from Subsection 2.4.2 in this setting and obtain both some structural results about such groups and their subgroups but also decidability results. The class of automaton groups of bounded activity is interesting for two reasons: first, many of the well studied automaton groups are in fact of bounded activity (see e. g. the introduction of [Bon+13]) and, second, many problems that are undecidable in the general case are decidable for automata of bounded activity. One example of this is that the order problem is undecidable for automaton groups [Gil18] even in the contracting case [BM20] but it is decidable for automaton groups [Bon+13] and semigroups [Bar+18] of bounded activity. Similarly, the conjugacy problem is undecidable for general automaton groups [ŠV12] but decidable within the group of regular tree automorphisms of bounded activity [Bon+13]. We already discussed the situation for the finiteness problem in the general case in Section 2.3 and we will see in this subsection that the finiteness problem for automaton groups of bounded activity and for their finitely generated subgroups is decidable. Finally, we will see that the problem of checking whether a given $\mathscr{G}$-automaton acts spherically transitive (an open problem in the general case [GNS00, 7.2 e) and f)]) can be decided for (finitely generated subgroups of) automaton groups of bounded activity and we also obtain decidability results regarding torsion and torsion-freeness for the dual automaton.

Activity. Let $\mathcal{T} = (Q, \Sigma, \delta)$ be an $\mathscr{S}$-automaton and let $q \in Q$. We define

$$\begin{aligned} A_q : \mathbb{N} &\to \Sigma^* \\ n &\mapsto \{v \mid \exists u \in \Sigma^n : v = q \circ u \text{ (defined, in particular) and} \\ &\qquad\qquad q \cdot u \neq_{\mathcal{T}} \varepsilon\}. \end{aligned}$$

The *activity of* q is the map[86] $\alpha_q : \mathbb{N} \to \mathbb{N}$ with $\alpha_q(n) = |A_q(n)|$ and q has *bounded activity* if there is a constant K such that $\alpha_q(n) \leq K$ for all $n \in \mathbb{N}$. The *activity of* $\mathcal{T}$ is $\alpha_{\mathcal{T}} : \mathbb{N} \to \mathbb{N}$ with[87] $\alpha_{\mathcal{T}}(n) = \sum_{q \in Q} \alpha_q(n)$ and $\mathcal{T}$ has *bounded activity* if there is some constant K such that $\alpha_{\mathcal{T}}(n) \leq K$ for all $n \in \mathbb{N}$. Note that the latter is equivalent to all states of $\mathcal{T}$ having bounded activity.

It is not difficult to see that pq (as a state of $\mathcal{T}^2$) has bounded activity if p and q have bounded activity (as states of $\mathcal{T}$) [Bar+18, Lemma 3.1]. If $\mathcal{T}$ is a $\mathscr{G}$-automaton, then it

[86] Usually, we use α for ω-words but, here, we also use it for the activity.

[87] We could define the activity of an automaton also by taking the maximum instead of the sum. However, taking the latter will make our life a bit easier below.

is also easy to see that $\mathcal{T}$ has bounded activity if and only if $\overline{\mathcal{T}}$ has; in fact, $q \in Q$ has bounded activity if and only if $\bar{q}$ has.

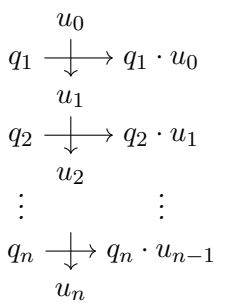
Remark 2.4.3.1. The notion of activity originally used by Sidki [Sid00] to define the polynomial and exponential activity hierarchies was a bit different to the one presented here. He only considered groups and defined $\alpha_q(n)$ as the cardinality of the set

$$\{u \in \Sigma^n \mid q \cdot u \neq \mathbb{1} \text{ in } \mathscr{G}(\mathcal{T})\}.$$

Here, we follow the definition of Bartholdi, Godin, Klimann and Picantin [Bar+18], who extended the notion to automaton semigroups. Clearly, the two definitions coincide in the case that $\mathcal{T}$ is a $\mathscr{G}$-automaton.

Directly from the definition of bounded activity, we obtain the following fact.

The situation as a cross diagram:



Fact 2.4.3.2. *Let $\mathcal{T} = (Q, \Sigma, \delta)$ be an $\mathscr{S}$-automaton whose activity is bounded by a constant K. Furthermore, let $u_0 \in \Sigma^*$ and $q_1, \ldots, q_n \in Q$ such that $q_n \ldots q_1 \circ u_0$ is defined and all $u_i = q_i \ldots q_1 \circ u_0$ with $1 \leq i \leq n$ are pairwise distinct. Then, $q_n \ldots q_1 \cdot u_0$ contains at most K states whose partial action is not the identity.*

Proof. Every u_i is an element of A_{q_i} if $q_i \cdot u_{i-1} \neq_{\mathcal{T}} \varepsilon$. However, $\bigcup_{q \in Q} A_q$ contains at most K elements. □

From now on, we will mostly be dealing with $\mathscr{G}$-automata and we, therefore, continue with the notational conventions of Subsection 2.4.2 and write $\mathrm{Stab}_{\boldsymbol{P}}(w)$ instead of $\widetilde{\boldsymbol{P}}^* \cap \mathrm{Stab}^{\mathbb{1}}_{\widetilde{\mathcal{T}}}(w)$ for $\boldsymbol{P} \subseteq \widetilde{Q}^*$ when the $\mathscr{G}$-automaton $\mathcal{T} = (Q, \Sigma, \delta)$ is clear from the context.

The previous fact allows us to improve the upper bound of Lemma 2.4.2.3 to a constant if the activity of the $\mathscr{G}$-automaton is bounded. This meas that all shifted stabilizers are generated by state sequences whose length is bounded by a constant.

Lemma 2.4.3.3. *Let $\mathcal{T} = (Q, \Sigma, \delta)$ be $\mathscr{G}$-automaton of bounded activity and let $P \subseteq \widetilde{Q}$. Then, we have:*

$$\exists K > 0 \, \forall w \in \Sigma^* \, \exists D \subseteq \widetilde{Q}^{\leq K} : \widetilde{D}^* = \mathrm{Stab}_P(w) \cdot w \text{ in } \mathscr{G}(\mathcal{T}).$$

Furthermore, the subset D generating $\mathrm{Stab}_P(w) \cdot w$ in $\mathscr{G}(\mathcal{T})$ can be computed from $\mathcal{T}$ and w.

Proof. Since $\mathcal{T}$ has bounded activity, so has $\widetilde{\mathcal{T}}$ and we let K' be a constant bounding the activity of $\widetilde{\mathcal{T}}$.

Let $w \in \Sigma^*$ be arbitrary. We fix some spanning tree in the $\widetilde{P}^*$-orbital graph $\widetilde{P}^* \circ w$ and choose C in the same way as in the proof of Lemma 2.4.2.3. Then, C generates $\mathrm{Stab}_P(w)$ in $\mathscr{G}(\mathcal{T})$ and, therefore, $C \cdot w$ generates $\mathrm{Stab}_P(w) \cdot w$ in $\mathscr{G}(\mathcal{T})$ (as a subgroup). We will show that $\boldsymbol{c} \cdot w$ for $\boldsymbol{c} \in C$ is equal to a computable element from $\widetilde{Q}^*$ of length at most $K = 2K' + 1$ in $\mathscr{G}(\mathcal{T})$, which shows the lemma (since we can compute the orbital graph $\widetilde{P}^* \circ w$, a spanning tree therein and the corresponding set C).

Let $\boldsymbol{c} \in C$ be arbitrary. By the construction of C, we have $\boldsymbol{c} = t'_n \dots t'_1 p t_m \dots t_1$ where $\boldsymbol{t} = t_m \dots t_1 \in \widetilde{P}^*$ belongs to a simple path from w to u (along the spanning tree) in $\widetilde{P}^* \circ w$, $p \in P$ is the label of a single edge $v \xleftarrow{p} u$ and $\boldsymbol{t}' = t'_n \dots t'_1 \in \widetilde{P}^*$ belongs to a simple path from v back to w. Therefore, all $w_i = t_i \dots t_1 \circ w$ with $1 \leq i \leq m$ are pairwise distinct and the same holds for all $w'_j = t'_j \dots t'_1 \circ v$ with $1 \leq j \leq n$. We obtain that $\boldsymbol{t} \cdot w$ contains at most K' states different to the neutral element in $\mathscr{G}(\mathcal{T})$ by Fact 2.4.3.2 and, in the same way, that $\boldsymbol{t}' \cdot v$ contains at most K' such states as well. Thus, $\boldsymbol{c} \cdot w = (\boldsymbol{t}' \cdot v)(p \cdot u)(\boldsymbol{t} \cdot w)$ is equal to an element of length at most $K = 2K' + 1$ in $\mathscr{G}(\mathcal{T})$ and, since we can test equality to the identity (i. e. solve the word problem, see Section 2.1), we can compute this element. □

We will use Lemma 2.4.3.3 to describe the $\boldsymbol{P}^*$-orbit size of a word by a weighted acceptor.

Weighted Acceptor. A (finite) *weighted acceptor*[88] is a tuple $\mathcal{A} = (Q, \Sigma, \delta, q_0)$ where Q is a finite set of *states*, Σ is an alphabet, $\delta \subseteq Q \times \Sigma \times \mathbb{N} \times Q$ is a set of *weighted transitions* and $q_0 \in Q$ is the *initial state*. The weighted transition (p, a, γ, q) goes *from* state p *to* state q and has *weight* γ. It is *labeled* by a. As is the case with (our usual) automata, we use the graphical notation $p \xrightarrow[\gamma]{a} q$ for the weighted transition (p, a, γ, q). A *run* of the weighted acceptor $\mathcal{A} = (Q, \Sigma, \delta, q_0)$ is a sequence

$$p_0 \xrightarrow[\gamma_1]{a_1} p_1 \xrightarrow[\gamma_2]{a_2} \dots \xrightarrow[\gamma_n]{a_n} p_n$$

of weighted transitions $p_{i-1} \xrightarrow[\gamma_i]{a_i} p_i \in \delta$ with $1 \leq i \leq n$. It is labeled by $a_1 \dots a_n$ and its *weight* is $\prod_{i=1}^{n} \gamma_i$. The run is *initial* if p_0 is the initial state of $\mathcal{A}$. Clearly, the idea of a run can be extended to infinite runs and, thus, to ω-words. Here, the weight of a run can possibly be infinite.

We will only consider *deterministic* and *complete* weighted acceptors, i. e. we have

$$d_{p,a} = \left| \left\{ p \xrightarrow[\gamma]{a} q \in \delta \;\middle|\; \gamma \in \mathbb{N}, q \in Q \right\} \right| = 1$$

for all $p \in Q$ and $a \in \Sigma$, and will not explicitly mention these properties. Thus, a weighted acceptor $\mathcal{A} = (Q, \Sigma, \delta)$ has exactly one initial run for every word w over Σ. It *accepts* the function $f : \Sigma^\infty \to \mathbb{N} \cup \{\infty\}$ which maps a word $w \in \Sigma^\infty$ to the weight of the unique initial run labeled by w.

Our goal is to describe – or, more precisely, compute – a weighted acceptor that accepts the function mapping a word to its $\boldsymbol{P}^*$-orbit size (for a finite set $\boldsymbol{P}$ of state sequences). We will use the following fact to show that our acceptor is well-defined.

Fact 2.4.3.4. *Let $\mathcal{T} = (Q, \Sigma, \delta)$ be an $\mathscr{S}$-automaton, $P \subseteq Q$. For all $w, w' \in \Sigma^*$ and $a \in \Sigma$, we have*

$$\begin{aligned} &\mathrm{Stab}_P(w) \cdot w = \mathrm{Stab}_P(w') \cdot w' \text{ in } \mathscr{S}(\mathcal{T}) \\ &\qquad\qquad \Longrightarrow \mathrm{Stab}_P(wa) \cdot wa = \mathrm{Stab}_P(w'a) \cdot w'a \text{ in } \mathscr{S}(\mathcal{T}) \end{aligned}$$

where we write $\mathrm{Stab}_P(u)$ instead of $P^* \cap \mathrm{Stab}_{\mathcal{T}}(u)$ for finite words $u \in \Sigma^*$.

[88] We use the term "acceptor" here again instead of the more common term "automaton" because we use the latter to refer to transducers. Also our notion of a weighted acceptor/automaton is only a special case of the usual, more general notion.

Proof. Let $\boldsymbol{q} \in \mathrm{Stab}_P(wa) \cdot wa$. By symmetry, we only have to show that there is some $\boldsymbol{q}' \in \mathrm{Stab}_P(w'a) \cdot w'a$ with $\boldsymbol{q} = \boldsymbol{q}'$ in $\mathscr{S}(\mathcal{T})$. There has to be some $\boldsymbol{p} \in \mathrm{Stab}_P(wa)$ with $\boldsymbol{p} \cdot wa = \boldsymbol{q}$. We have $\boldsymbol{p} \in \mathrm{Stab}_P(wa) \subseteq \mathrm{Stab}_P(w)$ and, therefore, $\boldsymbol{p} \cdot w \in \mathrm{Stab}_P(w) \cdot w$. By the hypothesis, there has to be some $\boldsymbol{p}' \in \mathrm{Stab}_P(w')$ with $\boldsymbol{p}' \cdot w' = \boldsymbol{p} \cdot w$ in $\mathscr{S}(\mathcal{T})$. In particular, we have $\boldsymbol{p}' \cdot w' \circ a = \boldsymbol{p} \cdot w \circ a = a$ and, therefore, $\boldsymbol{p}' \circ w'a = w'a$ and $\boldsymbol{p}' \in \mathrm{Stab}_P(w'a)$. Finally, we get $\boldsymbol{q}' = \boldsymbol{p}' \cdot w'a = \boldsymbol{p} \cdot wa = \boldsymbol{q}$ in $\mathscr{S}(\mathcal{T})$ by Fact 0.3.1.1. □

Theorem 2.4.3.5. *Let $\mathcal{T} = (Q, \Sigma, \delta)$ be a $\mathscr{G}$-automaton of bounded activity and let $\boldsymbol{P} \subseteq \widetilde{Q}^*$ be finite. Then, it is possible to compute a weighted acceptor from $\mathcal{T}$ that accepts the function*

$$\begin{aligned} \Sigma^\infty &\to \mathbb{N} \cup \{\infty\} \\ w &\mapsto |\boldsymbol{P}^* \circ w|. \end{aligned}$$

Proof. We may assume $\boldsymbol{P} = P \subseteq \widetilde{Q}$ as we can replace $\mathcal{T}$ by a union with suitable powers of $\mathcal{T}$ (these powers and the union are also of bounded activity and can be computed). Let K be the constant from Lemma 2.4.3.3 belonging to (the possibly new) $\mathcal{T} = (Q, \Sigma, \delta)$. Now, for every $w \in \Sigma^*$, we can compute[89] $D(w) \subseteq \widetilde{Q}^{\leq K}$ such that $D(w)$ generates $\mathrm{Stab}_P(w) \cdot w$ (as a subgroup) in $\mathscr{G}(\mathcal{T})$.

[89] Note that it is not necessary to compute K for this.

We will define the weighted acceptor over the alphabet Σ via a saturation process. We start with the single state $D(\varepsilon)$, which we choose as the dedicated initial state. As long as we have a state $D(w)$ without outgoing transitions, we add the transition $D(w) \xrightarrow[\gamma]{a} D(wa)$ for every $a \in \Sigma$ whose weight γ we define later. Adding such a transition possibly also adds a new state $D(wa)$. However, since all $D(w)$ are subsets of $\widetilde{Q}^{\leq K}$, the process has to terminate. Note that the resulting acceptor is complete. Finally, we define the weight γ of a transition $C \xrightarrow[\gamma]{a} D$ as the cardinality of $\widetilde{C}^* \circ a = C^* \circ a$ (see Fact 0.3.2.2). This cardinality can be computed by its own saturation process.

We have no guarantee that $D(w) = D(w')$ implies $D(wa) = D(w'a)$. However, we can guarantee some equality in the group. Let $w = a_1 \dots a_n \in \Sigma^*$ with $a_1, \dots, a_n \in \Sigma$ be some word and consider its (unique and existing) initial run

$$D_0 \xrightarrow[\gamma_1]{a_1} D_1 \ \dots \xrightarrow[\gamma_n]{a_n} D_n.$$

We claim that we have $\widetilde{D}_i^* = \mathrm{Stab}_P(w_i) \cdot w_i$ in $\mathscr{G}(\mathcal{T})$ for all $0 \leq i \leq n$ where $w_i = a_1 \dots a_i$ and that $\gamma_1 \dots \gamma_i$ is the orbit size $|P^* \circ w_i|$. Clearly, this is true for $i = 0$. For $i > 0$, there is some $w' \in \Sigma^*$ such that $D_{i-1} = D(w')$ and $D_i = D(w'a_i)$ by construction of the acceptor. By definition, we have that $D(w')$ generates $\mathrm{Stab}_P(w') \cdot w'$ in $\mathscr{G}(\mathcal{T})$ (as a subgroup) and, by induction, we obtain $\mathrm{Stab}_P(w') \cdot w' = \mathrm{Stab}_P(w_{i-1}) \cdot w_{i-1}$ in $\mathscr{G}(\mathcal{T})$. By Fact 2.4.3.4, this implies that $\mathrm{Stab}_P(w'a_i) \cdot w'a_i$, which is generated by $D(w'a_i) = D_i$ in $\mathscr{G}(\mathcal{T})$, is equal to $\mathrm{Stab}_P(w_{i-1}a_i) \cdot w_{i-1}a_i = \mathrm{Stab}_P(w_i) \cdot w_i$ in $\mathscr{G}(\mathcal{T})$. This establishes the first part of the claim. For the second part, we observe that we have $\widetilde{D}_{i-1}^* = \mathrm{Stab}_P(w_{i-1}) \cdot w_{i-1}$ in $\mathscr{G}(\mathcal{T})$. From Lemma 2.4.2.1, we obtain

$$\begin{aligned} |P^* \circ w_i| = |P^* \circ w_{i-1}a_i| &= |P^* \circ w_{i-1}| \cdot |\,\mathrm{Stab}_P(w_{i-1}) \cdot w_{i-1} \circ a_i| \\ &= |P^* \circ w_{i-1}| \cdot |\widetilde{D}_{i-1}^* \circ a_i| = \gamma_1 \dots \gamma_{i-1} \cdot \gamma_i \end{aligned}$$

where the equality $|P^* \circ w_{i-1}| = \gamma_1 \dots \gamma_{i-1}$ follows from induction. □

Theorem 2.4.3.5 has some interesting consequences. The first one is that we can define those transitions in the weighted acceptor with weight larger than one as *accepting*. We consider an ω-word as *accepted* by the acceptor if its initial run contains infinitely many accepting transitions. In this way, we obtain a deterministic Büchi acceptor for the language of ω-words with an infinite $\boldsymbol{P}^*$-orbit. We will not define Büchi acceptors(/automata) formally but refer the reader to Perrin and Pin's book [PP04] for more details.

Corollary 2.4.3.6 (compare to [2, Theorem 1.2])**.** *Let $\mathcal{T} = (Q, \Sigma, \delta)$ be a $\mathscr{G}$-automaton of bounded activity and let $\boldsymbol{P} \subseteq \widetilde{Q}^*$ be finite. Then, it is possible to compute a deterministic Büchi acceptor from $\mathcal{T}$ that accepts the set*

$$\{\alpha \in \Sigma^\omega \mid |\boldsymbol{P}^* \circ \alpha| = \infty\}$$

of ω-words with infinite $\boldsymbol{P}^*$-orbit

We also obtain that the set of $\boldsymbol{P}^*$-orbit expandable words for a finite set $\boldsymbol{P} \subseteq \widetilde{Q}^*$ with respect to a $\mathscr{G}$-automaton $\mathcal{T} = (Q, \Sigma, \delta)$ of bounded activity is recognized by a deterministic finite acceptor (DFA)[90] (i. e. it is *regular*): we mark those states as final from which a transition whose weight is larger than one can be reached (in the weighted acceptor from Theorem 2.4.3.5 where we forget the weight afterwards).

[90] We will also not define DFAs formally but refer the reader to a standard textbook on automaton theory (such as [HU79]).

Because the states of the weighted acceptor from Theorem 2.4.3.5 are subsets of $\widetilde{Q}^{\leq K}$ (where K is the constant from Lemma 2.4.3.3 with respect to the $\mathscr{G}$-automaton $\mathcal{T} = (Q, \Sigma, \delta)$), we obtain that there can be at most $2^{|\widetilde{Q}^{\leq K}|} < 2^{|\widetilde{Q}|^{K+1}}$ such states. Thus, if we can reach a weighted transition with weight larger than one from some state, we can already reach it in less than $2^{|\widetilde{Q}|^{K+1}}$ steps, which shows the following improvement over Corollary 2.4.2.5 (and Corollary 2.4.1.4).

Corollary 2.4.3.7. *Let $\mathcal{T} = (Q, \Sigma, \delta)$ be a $\mathscr{G}$-automaton of bounded activity and let $\boldsymbol{P} \subseteq \widetilde{Q}^*$ be finite. Then, there is a constant M such that a word $w \in \Sigma^*$ is $\boldsymbol{P}^*$-orbit expandable if and only if it is already $\boldsymbol{P}^*$-orbit expandable by some $x \in \Sigma^*$ with*

$$|x| < 2^{(2|Q|)^M}.$$

Proof. If we have $Q = \{q\}$ for a single state q, the action of q must be a letter-wise permutation of order $|\mathscr{G}(\mathcal{T})|$ and we obtain a maximal $\boldsymbol{P}^*$-orbit size by appending a suffix which contains all letters in Σ. Thus, we only have to choose M with $|\Sigma| < 2^{2^M}$.

Thus, assume $|Q| > 1$. Only one direction is non-trivial and we let $w \in \Sigma^*$ be $\boldsymbol{P}^*$-orbit expandable. Furthermore, let $N = \max\{|\boldsymbol{p}| \mid \boldsymbol{p} \in \boldsymbol{P}\}$ and $\mathcal{T}' = (Q', \Sigma, \delta') = \biguplus_{i=1}^{N} \mathcal{T}^i$. Then, we have $\boldsymbol{P} \subseteq \widetilde{Q'}$ and

$$|Q'| = \sum_{i=1}^{N} |Q|^i = \frac{|Q|^{N+1} - 1}{|Q| - 1} - 1 < |Q|^{N+1}.$$

We let K be the constant from Lemma 2.4.3.3 with respect to $\mathcal{T}'$ and obtain from the size

of the automaton from Theorem 2.4.3.5 (with respect to $\mathcal{T}'$) that there is some $x \in \Sigma^*$ with $|\boldsymbol{P}^* \circ w| < |\boldsymbol{P}^* \circ wx|$ and

$$|x| \leq 2^{\left|(\widetilde{Q}')^{\leq K}\right|} < 2^{(2|Q'|)^{K+1}} < 2^{(2|Q|)^{(N+1)(K+1)}}.$$

Thus, we can choose $M = (N+1)(K+1)$. □

The next application of Theorem 2.4.3.5 is again a structural result: an infinite automaton group of bounded activity yields a periodic word with an infinite orbit. In fact, the periodic word can even be computed. In addition, an infinite finitely generated subgroup of an automaton group of bounded activity still admits an ultimately periodic word with an infinite orbit.

Corollary 2.4.3.8. *Let $\mathcal{T} = (Q, \Sigma, \delta)$ be a $\mathscr{G}$-automaton of bounded activity. Then, the statements*

1. $|\mathscr{G}(\mathcal{T})| = \infty$
2. $\exists u \in \Sigma^*, v \in \Sigma^+ : |Q^* \circ uv^\omega| = \infty$
3. $\exists v \in \Sigma^+ : |Q^* \circ v^\omega| = \infty$
4. $\exists w \in \Sigma^+ : w$ *has no torsion in* $\mathscr{S}(\partial\mathcal{T})$

are equivalent. Furthermore, the subgroup generated by a finite set $\boldsymbol{P} \subseteq \widetilde{Q}^*$ in $\mathscr{G}(\mathcal{T})$ is infinite if and only if

$$\exists u \in \Sigma^*, v \in \Sigma^+ : |\boldsymbol{P}^* \circ uv^\omega| = \infty.$$

The current author would like to thank Dominik Francoeur for pointing out the implication 2 $\implies$ 3.

Proof. The implications 3 $\implies$ 2 and 2 $\implies$ 1 (as well as the corresponding direction of the second statement) are clear. The implication 2 $\implies$ 3 follows from the inclusion $Q^* \circ uv^\omega \subseteq \Sigma^{|u|}(Q^* \circ v^\omega)$ and the equivalence 4 $\iff$ 3 follows from Theorem 1.4.2.2.

Finally, the implication 1 $\implies$ 2 is a special case of the (other direction of the) second statement. To see the latter, observe that there is some $\alpha \in \Sigma^\omega$ with an infinite $\boldsymbol{P}^*$-orbit $\boldsymbol{P}^* \circ \alpha$ if the subgroup $\widetilde{\boldsymbol{P}}^*$ in $\mathscr{G}(\mathcal{T})$ is infinite by Theorem 1.4.1.13 (and Fact 0.3.2.2, where, to apply the former, we have to assume that $\boldsymbol{P}$ is a subset of $\widetilde{Q}$ again without loss of generality). Therefore, the Büchi acceptor from Corollary 2.4.3.6 accepts at least one word. Thus, there must be a cycle with an accepting state/transition reachable from the initial state and the acceptor must accept an ultimately periodic word. □

Remark 2.4.3.9. We can obtain the previous result without all the machinery behind Corollary 1.4.1.14. If an automaton group (or one of its finitely generated subgroups) is infinite, then we can find arbitrarily long words with arbitrarily large orbits. These also yield a reachable cycle containing a transition with weight larger than one in the weighted acceptor from Theorem 2.4.3.5.

Other applications of Theorem 2.4.3.5 concern decidability results. The first example of this is that the finiteness problem is decidable for the class of automaton groups of bounded activity and their finitely generated subgroups (the former is a special case of the latter). We have already discussed the situation of the finiteness problem for general automaton groups and semigroups in Section 2.3.

Corollary 2.4.3.10 (compare to [2, Theorem 1.1])**.** *The finiteness problem for finitely generated subgroups of automaton groups of bounded activity*[91]

Input: *a $\mathscr{G}$-automaton $\mathcal{T}$ of bounded activity and a finite set $\boldsymbol{P} \subseteq \widetilde{Q}^*$*

Question: *is the subgroup generated by $\boldsymbol{P}$ in $\mathscr{G}(\mathcal{T})$ infinite?*

is decidable.

[91] Note that it is decidable whether a given $\mathscr{G}$-automaton has bounded activity: this is the case if and only if no cycle (containing a non-identity state) is reachable from another one [Sid00, Corollary 14].

Proof. First, we replace the input $\mathscr{G}$-automaton by a suitable union $\mathcal{T} = (Q, \Sigma, \delta)$ of the original $\mathscr{G}$-automaton with some of its powers so that we obtain $\boldsymbol{P} = P \subseteq \widetilde{Q}$. This is computable, does not change the group and $\mathcal{T}$ remains to be of bounded activity.

By Corollary 1.4.1.14 (and Fact 0.3.2.2), the subgroup $\widetilde{P}^*$ in $\mathscr{G}(\mathcal{T})$ is now infinite if and only if there is some $\alpha \in \Sigma^\omega$ with $|P^* \circ \alpha| = \infty$. The latter can be decided by computing the Büchi automaton mentioned in Corollary 2.4.3.6 and checking whether it contains a cycle reachable from the initial state with at least one accepting transition/state. □

Similarly, we obtain that the torsion problem and the torsion-freeness problem for reversible input $\mathscr{S}$-automaton whose dual is of bounded activity are decidable.

Corollary 2.4.3.11. *The problems*

Input: *a reversible $\mathscr{S}$-automaton $\mathcal{T} = (Q, \Sigma, \delta)$ whose dual $\partial\mathcal{T}$ is of bounded activity*

Question: *$\exists \boldsymbol{q} \in Q^+ : \boldsymbol{q}$ has no torsion in $\mathscr{S}(\mathcal{T})$?*

and[92]

Input: *a reversible $\mathscr{S}$-automaton $\mathcal{T} = (Q, \Sigma, \delta)$ whose dual $\partial\mathcal{T}$ is of bounded activity*

Question: *$\exists \boldsymbol{q} \in Q^+ : \boldsymbol{q}$ has torsion in $\mathscr{S}(\mathcal{T})$?*

are decidable.

[92] The corresponding problem for general $\mathscr{S}$-automata is undecidable by Corollary 2.2.2.6.

Proof. By Corollary 2.4.3.8, the first problem is equivalent to the finiteness problem for $\partial\mathcal{T}$ and, thus, decidable by Corollary 2.4.3.10.

Similarly, we obtain from Theorem 1.4.2.3 that $\mathscr{S}(\mathcal{T})$ contains an element of torsion if and only if there is a periodic ω-word with finite orbit under that action of the dual. The latter can be decided from the acceptor in Theorem 2.4.3.5 (or Corollary 2.4.3.6). □

The next example of a decidability result following from Theorem 2.4.3.5 is about spherical transitivity. A set $K \subseteq Q^*$ for an $\mathscr{S}$-automaton $\mathcal{T} = (Q, \Sigma, \delta)$ acts *spherically transitive* (or *level transitive*) if, for every $n \in \mathbb{N}$ and every $u, v \in \Sigma^n$, there is some $\boldsymbol{q} \in K$ with $v = \boldsymbol{q} \circ u$. An alternative formulation is that every word from Σ^n has a K-orbit of size $|\Sigma|^n$. If we have a $\mathscr{G}$-automaton $\mathcal{T} = (Q, \Sigma, \delta)$ and K is of the form $\boldsymbol{P}^*$ for a finite set $\boldsymbol{P} \subseteq \widetilde{Q}$, K acts spherically transitive if and only if every transition of the weighted acceptor from Theorem 2.4.3.5 has weight $|\Sigma|$ and we obtain the following corollary.[93]

[93] The corresponding problem in the general case (even without subgroups) is an open problem [GNS00, 7.2 e) and f)].

Corollary 2.4.3.12 (compare to [2, Corollary 1.4])**.** *The problem*

Input: *a $\mathscr{G}$-automaton $\mathcal{T} = (Q, \Sigma, \delta)$ of bounded activity and a finite set $\boldsymbol{P} \subseteq \widetilde{Q}^*$*

Question: *is the action of $\boldsymbol{P}^*$ spherically transitive?*

is decidable.

Bibliography

Cited Publications of the Author (chronologically)

[1] Daniele D'Angeli, Emanuele Rodaro, and Jan Philipp Wächter. "On the complexity of the word problem for automaton semigroups and automaton groups." *Advances in Applied Mathematics* 90 (2017), pp. 160–187. ISSN: 0196-8858. DOI: 10.1016/j.aam.2017.05.008. arXiv: 1611.09541 [cs.FL].

[2] Ievgen Bondarenko and Jan Philipp Wächter. "On Orbits and the Finiteness of Bounded Automaton Groups." *arXiv preprint* (2019). arXiv: 1912.06897 [math.GR]. Accepted for the special issue on "Semigroups and Groups, Automata, Logics" (SandGAL 2019) appearing in *International Journal of Algebra and Computation*.

[3] Daniele D'Angeli, Emanuele Rodaro, and Jan Philipp Wächter. "Automaton semigroups and groups: On the undecidability of problems related to freeness and finiteness." *Israel Journal of Mathematics* 237 (1 2020), pp. 15–52. ISSN: 1565-8511. DOI: 10.1007/s11856-020-1972-5. arXiv: 1712.07408 [cs.FL]. Corrected in [6].

[4] Daniele D'Angeli, Emanuele Rodaro, and Jan Philipp Wächter. "Orbit Expandability of Automaton Semigroups and Groups." *Theoretical Computer Science* 809 (2020), pp. 418–429. ISSN: 0304-3975. DOI: 10.1016/j.tcs.2019.12.037. arXiv: 1812.07359 [cs.FL].

[5] Jan Philipp Wächter and Armin Weiß. "An Automaton Group with PSpace-Complete Word Problem." In: 37th International Symposium on Theoretical Aspects of Computer Science (STACS 2020). (Montpellier, France, Mar. 10–13, 2020). Ed. by Christophe Paul and Markus Bläser. Vol. 154. Leibniz International Proceedings in Informatics (LIPIcs). Dagstuhl, Germany: Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2020, 6:1–6:17. ISBN: 978-3-95977-140-5. DOI: 10.4230/LIPIcs.STACS.2020.6. arXiv: 1906.03424 [cs.FL]. Submitted by invitation to the special issue appearing in *Theory of Computing Systems*.

[6] Daniele D'Angeli, Emanuele Rodaro, and Jan Philipp Wächter. "Erratum to 'Automaton Semigroups and Groups: On the Undecidability of Problems Related to Freeness and Finiteness'." *arXiv preprint* (2020), E-1–E-5. arXiv: 1712.07408v3 [cs.FL]. Accepted by the *Israel Journal of Mathematics*. Erratum to [3].

[7] Daniele D'Angeli, Dominik Francoeur, Emanuele Rodaro, and Jan Philipp Wächter. "Infinite automaton semigroups and groups have infinite orbits." *Journal of Algebra* 553 (2020), pp. 119–137. ISSN: 0021-8693. DOI: 10.1016/j.jalgebra.2020.02.014. arXiv: 1903.00222 [cs.FL].

[8] Daniele D'Angeli, Dominik Francoeur, Emanuele Rodaro, and Jan Philipp Wächter. "On the Orbits of Automaton Semigroups and Groups." *arXiv preprint* (2020). arXiv: 2007.10273 [cs.FL]. Accepted by *Algebra and Discrete Mathematics*.

[9] Daniele D'Angeli, Emanuele Rodaro, and Jan Philipp Wächter. "On the structure theory of partial automaton semigroups." *Semigroup Forum* 101 (2020), pp. 51–76. ISSN: 1432-2137. DOI: 10.1007/s00233-020-10114-5. arXiv: 1811.09420 [cs.FL].

Other Publications of the Author (chronologically)

- Jan Philipp Wächter. "Kaskadenzerlegung spezieller Automatenklassen." German. Advisor: Manfred Kufleitner. Examiner: Volker Diekert. Studienarbeit (Senior Thesis). Universität Stuttgart, 2013. DOI: 10.18419/opus-3170.
- Jan Philipp Wächter. "Das Wortproblem für Omega-Terme über Zweivariablenlogik." German. Advisor: Manfred Kufleitner. Examiner: Volker Diekert. Diplomarbeit (Diploma Thesis). Universität Stuttgart, 2014. DOI: 10.18419/opus-3431.
- Manfred Kufleitner and Jan Philipp Wächter. "Two-Variable Ehrenfeucht-Fraisse Games over Omega-Terms." *arXiv preprint* (2014). arXiv: 1411.0593 [cs.LO].
- Manfred Kufleitner and Jan Philipp Wächter. "The Word Problem for Omega-Terms over the Trotter-Weil Hierarchy." *Theory of Computing Systems* 62 (3 2018): *Special Issue on Computer Science in Russia 2016*. Ed. by Alexander S. Kulikov and Gerhard J. Woeginger, pp. 682–738. ISSN: 1433-0490. DOI: 10.1007/s00224-017-9763-z. arXiv: 1509.05364 [cs.FL].

Other References (alphabetically)

[AIM07] *Problem List for the AIM Workshop "Selfsimilar groups and conformal dynamics" (American Institute of Mathematics, San Jose, California, June 5–9, 2006)*. 2007. URL: https://aimath.org/WWN/selfsimgroups/selfsimgroups.pdf (visited on 06/11/2020).

[Akh+12] Ali Akhavi, Ines Klimann, Sylvain Lombardy, Jean Mairesse, and Matthieu Picantin. "On the Finiteness Problem for Automaton (Semi)Groups." *International Journal of Algebra and Computation* 22.06 (2012), p. 1250052. DOI: 10.1142/S021819671250052X. arXiv: 1105.4725 [cs.FL].

[Bar89] David A. Mix Barrington. "Bounded-Width Polynomial-Size Branching Programs Recognize Exactly Those Languages in NC^1." *Journal of Computer and System Sciences* 38.1 (1989), pp. 150–164. DOI: 10.1016/0022-0000(89)90037-8.

[Bar+20] Laurent Bartholdi, Michael Figelius, Markus Lohrey, and Armin Weiß. "Groups with ALogTime-hard word problems and PSpace-complete circuit value problems." Leibniz International Proceedings in Informatics (LIPIcs) 169 (2020). Ed. by Shubhangi Saraf, 29:1–29:29. ISSN: 1868-8969. DOI: 10.4230/LIPIcs.CCC.2020.29. arXiv: 1909.13781 [math.GR].

[Bar+18] Laurent Bartholdi, Thibault Godin, Ines Klimann, and Matthieu Picantin. "A New Hierarchy for Automaton Semigroups." In: *Implementation and Application of Automata*. 23rd International Conference on Implementation and Applications of Automata (CIAA 2018). (Charlottetown, Prince Edward Island, Canada, July 30–Aug. 2, 2018). Ed. by Cezar Câmpeanu. Vol. 10977. Lecture Notes in Computer Science. Cham: Springer, 2018, pp. 71–83. ISBN: 978-3-319-94812-6. DOI: 10.1007/978-3-319-94812-6_7. arXiv: 1803.09991 [cs.FL].

[BM20] Laurent Bartholdi and Ivan Mitrofanov. "The word and order problems for self-similar and automata groups." *Groups, Geometry, and Dynamics* 14 (2020), pp. 705–728. ISSN: 1661-7207. DOI: 10.4171/GGD/560. arXiv: 1710.10109 [math.GR].

[BP10] Paul C. Bell and Igor Potapov. "On the undecidability of the identity correspondence problem and its applications for word and matrix semigroups." *International Journal of Foundations of Computer Science* 21.06 (2010), pp. 963–978. DOI: 10.1142/S0129054110007660.

[Ber66] Robert Berger. *The Undecidability of the Domino Problem*. Memoirs of the American Mathematical Society 66. American Mathematical Society, 1966. ISBN: 978-0-821-81266-2.

[Bon12] Ievgen V. Bondarenko. "Growth of Schreier graphs of automaton groups." *Mathematische Annalen* 354 (2 2012), pp. 765–785. DOI: 10.1007/s00208-011-0757-x. arXiv: 1101.3200 [math.GR].

[Bon14] Ievgen V. Bondarenko. "The word problem in Hanoi Towers groups." *Algebra and Discrete Mathematics,* 17.2 (2014), pp. 248–255. URL: http://adm.luguniv.edu.ua/downloads/issues/2014/N2/adm-n2(2014)-5.pdf.

[Bon+13] Ievgen Bondarenko, Natalia V. Bondarenko, Said N. Sidki, and Flavia R. Zapata. "On the conjugacy problem for finite-state automorphisms of regular rooted trees." (with an appendix by Raphaël M. Jungers). *Groups, Geometry, and Dynamics* 7 (2 2013), pp. 232–355. DOI: 10.4171/GGD/184.

[BC15] Tara Brough and Alan J. Cain. "Automaton semigroup constructions." *Semigroup Forum* 90 (3 2015), pp. 763–774. ISSN: 1432-2137. DOI: 10.1007/s00233-014-9632-x.

[BC17] Tara Brough and Alan J. Cain. "Automaton semigroups: New constructions results and examples of non-automaton semigroups." *Theoretical Computer Science* 674 (2017), pp. 1–15. ISSN: 0304-3975. DOI: 10.1016/j.tcs.2017.02.003.

[BS98] Andrew M. Brunner and Said Sidki. "The generation of $GL(n, \mathbb{Z})$ by finite state automata." *International Journal of Algebra and Computation* 08.01 (1998), pp. 127–139. DOI: 10.1142/S0218196798000077.

[Bur02] William Burnside. "On an unsettled question in the theory of discontinuous groups." *The Quarterly Journal of Pure and Applied Mathematics* 33 (1902), pp. 230–238.

[Cai09] Alan J. Cain. "Automaton semigroups." *Theoretical Computer Science* 410.47 (2009), pp. 5022–5038. ISSN: 0304-3975. DOI: 10.1016/j.tcs.2009.07.054.

[Car+68] L. Carlitz, A. Wilansky, John Milnor, R. A. Struble, Neal Felsinger, J. M. S. Simoes, E. A. Power, R. E. Shafer, and R. E. Maas. "Advanced Problems: 5600-5609." *The American Mathematical Monthly* 75.6 (1968), pp. 685–687. ISSN: 00029890, 19300972. DOI: 10.2307/2313822.

[DR16] Daniele D'Angeli and Emanuele Rodaro. "Freeness of automaton groups vs boundary dynamics." *Journal of Algebra* 462 (2016), pp. 115–136. ISSN: 0021-8693. DOI: 10.1016/j.jalgebra.2016.05.015. arXiv: 1410.6097 [math.GR].

[Deh11] Max Dehn. "Über unendliche diskontinuierliche Gruppen." German. *Mathematische Annalen* 71 (1 1911), pp. 116–144. ISSN: 1432-1807. DOI: 10.1007/BF01456932.

[Fra18] Dominik Francoeur. "Infinite finitely generated automata semigroups have infinite orbits." *arXiv preprint* (2018). arXiv: 1801.10009 [math.GR].

[Gil14] Pierre Gillibert. "The finiteness problem for automaton semigroups is undecidable." *International Journal of Algebra and Computation* 24.01 (2014), pp. 1–9. DOI: 10.1142/S0218196714500015.

[Gil18] Pierre Gillibert. "An automaton group with undecidable order and Engel problems." *Journal of Algebra* 497 (2018), pp. 363–392. ISSN: 0021-8693. DOI: 10.1016/j.jalgebra.2017.11.049.

[GM05] Yair Glasner and Shahar Mozes. "Automata and Square Complexes." *Geometriae Dedicata* 111 (2005), pp. 43–64. ISSN: 1572-9168. DOI: 10.1007/s10711-004-1815-2. arXiv: math/0306259 [math.GR].

[Gri99] Rostislav I. Grigorchuk. "On the system of defining relations and the Schur multiplier of periodic groups generated by finite automata." In: Groups St Andrews 1997 in Bath. (University of Bath, July 26–Aug. 9, 1997). Ed. by Colin M. Campbell, Edmund F. Robertson, Nik Ruškuc, and Geoff C. Smith. Vol. 1. 2 vols. London Mathematical Society Lecture Note Series.

Cambridge University Press, 1999, pp. 290–317. ISBN: 978-1-107-36022-8. DOI: 10.1017/CBO9781107360228.021.

[GNS00] Rostislav I. Grigorchuk, Volodymyr V. Nekrashevych, and Vitaly I. Sushchanskĭi. "Automata, Dynamical Systems, and Groups." *Proceedings of the Steklov Institute of Mathematics* 231 (2000), pp. 128–203. Trans. of Rostislav I. Grigorchuk, Volodymyr V. Nekrashevych, and Vitaly I. Sushchanskiĭ. "Automata, Dynamical Systems, and Groups." Russian. In: *Dynamical Systems, Automata, and Infinite Groups. Collected Papers*. Vol. 231. Trudy Matematicheskogo Instituta Imeni V. A. Steklova. Rossiĭskaya Akademiya Nauk. Nauka, MAIK «Nauka/Inteperiodika», 2000, pp. 134–214.

[GP08] Rostislav I. Grigorchuk and Igor Pak. "Groups of intermediate growth: an introduction." *L'Enseignement Mathématique* 54 (3-4 2008), pp. 251–272. ISSN: 0013-8584. DOI: 10.5169/seals-109938.

[GŻ01] Rostislav I. Grigorchuk and Andrzej Żuk. "The Lamplighter Group as a Group Generated by a 2-state Automaton, and its Spectrum." *Geometriae Dedicata* 87 (2001), pp. 209–244. ISSN: 1572-9168. DOI: 10.1023/A:1012061801279.

[HU79] John E. Hopcroft and Jeffrey D. Ullman. *Introduction to Automata Theory, Languages, and Computation*. Addison-Wesley, 1979. 418 pp. ISBN: 978-0-201-02988-8.

[How95] John M. Howie. *Fundamentals of Semigroup Theory*. London Mathematical Society Monographs. Clarendon Press, 1995. 362 pp. ISBN: 978-0-19-851194-6.

[KP99] Jarkko J. Kari and Panos Papasoglu. "Deterministic Aperiodic Tile Sets." *Geometric & Functional Analysis GAFA* 9.2 (1999), pp. 353–369. ISSN: 1420-8970. DOI: 10.1007/s000390050090.

[Kli13] Ines Klimann. "The finiteness of a group generated by a 2-letter invertible-reversible Mealy automaton is decidable." In: 30th International Symposium on Theoretical Aspects of Computer Science (STACS 2013). (Kiel, Germany, Feb. 27–Mar. 2, 2013). Ed. by Natacha Portier and Thomas Wilke. Vol. 20. Leibniz International Proceedings in Informatics (LIPIcs). Dagstuhl, Germany: Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2013, pp. 502–513. ISBN: 978-3-939897-50-7. DOI: 10.4230/LIPIcs.STACS.2013.502.

[Kli16] Ines Klimann. "Automaton Semigroups: The Two-state Case." *Theory of Computing Systems* 58 (4 2016), pp. 664–680. ISSN: 1433-0490. DOI: 10.1007/s00224-014-9594-0.

[KMP12] Ines Klimann, Jean Mairesse, and Matthieu Picantin. "Implementing Computations in Automaton (Semi)groups." In: *Implementation and Application of Automata*. 17th International Conference on Implementation and Application of Automata (CIAA 2012). (Porto, Portugal, July 17–20, 2012). Ed. by Nelma Moreira and Rogério Reis. Vol. 7381. Lecture Notes in Computer

Science. Berlin, Heidelberg: Springer, 2012, pp. 240–252. ISBN: 978-3-642-31606-7. DOI: 10.1007/978-3-642-31606-7_21. arXiv: 1310.4856 [cs.FL].

[KPS18] Ines Klimann, Matthieu Picantin, and Dmytro Savchuk. "A Connected 3-State Reversible Mealy Automaton Cannot Generate an Infinite Burnside Group." *International Journal of Foundations of Computer Science* 29.02 (2018): *Special Issue: Developments in Language Theory (DLT 2015)*, pp. 297–314. DOI: 10.1142/S0129054118400087. arXiv: 1409.6142 [cs.FL].

[Koz77] Dexter Kozen. "Lower bounds for natural proof systems." In: 18th Annual Symposium on Foundations of Computer Science (SFCS 1977). (Providence, Rhode Island, USA, Oct. 31–Nov. 2, 1977). IEEE Computer Society Press, 1977, pp. 254–266. DOI: 10.1109/SFCS.1977.16.

[Loh14] Markus Lohrey. *The Compressed Word Problem for Groups*. SpringerBriefs in Mathematics. New York: Springer, 2014. ISBN: 978-1-4939-0747-2. DOI: 10.1007/978-1-4939-0748-9.

[Luk09] Ville Lukkarila. "The 4-way deterministic tiling problem is undecidable." *Theoretical Computer Science* 410.16 (2009), pp. 1516–1533. ISSN: 0304-3975. DOI: 10.1016/j.tcs.2008.12.006.

[Mak85] Gennadiĭ S. Makanin. "Decidability of the universal and positive theories of a free group." *Mathematics of the USSR-Izvestiya* 25 (1 1985), pp. 75–88. DOI: 10.1070/IM1985v025n01ABEH001269. Trans. of "Decidability of the universal and positive theories of a free group." Russian. *Izvestiya Akademii Nauk SSSR, Seriya Matematicheskaya* 48 (4 1984), pp. 735–749. ISSN: 1607-0046.

[Mal62] Anatolij I. Mal'cev. "On the equation $zxyx^{-1}y^{-1}z^{-1} = aba^{-1}b^{-1}$ in a free group." *Algebra i logika* 1.5 (1962), pp. 45–50. ISSN: 0373-9252.

[Mun74] Walter D. Munn. "Free Inverse Semigroups." *Proceedings of the London Mathematical Society* s3-29.3 (1974), pp. 385–404. DOI: 10.1112/plms/s3-29.3.385.

[Nek05] Volodymyr V. Nekrashevych. *Self-similar groups*. Vol. 117. Mathematical Surveys and Monographs. American Mathematical Society, Providence, RI, 2005. 231 pp. ISBN: 978-0-8218-3831-0. DOI: 10.1090/surv/117.

[Nek06] Volodymyr V. Nekrashevych. "Self-similar inverse semigroups and Smale spaces." *International Journal of Algebra and Computation* 16.5 (2006), pp. 849–874. ISSN: 0218-1967. DOI: 10.1142/S0218196706003153.

[OSS10] Andrij S. Olijnyk, Vitaly I. Sushchansky, and Janusz K. Słupik. "Inverse semigroups of partial automaton permutations." *International Journal of Algebra and Computation* 20.07 (2010), pp. 923–952. DOI: 10.1142/S0218196710005960.

[Pap94] Christos M. Papadimitriou. *Computational Complexity*. Addison-Wesley, 1994. 544 pp. ISBN: 978-0-201-53082-7.

[PP04] Dominique Perrin and Jean-Éric Pin. *Infinite words. Automata, Semigroups, Logic and Games*. Vol. 141. Pure and Applied Mathematics. Amsterdam: Elsevier, 2004. ISBN: 978-0-12-532111-2. URL: https://www.elsevier.com/books/infinite-words/perrin/978-0-12-532111-2.

[Pet84] Mario Petrich. *Inverse Semigroups*. Pure & Applied Mathematics. John Wiley & Sons Inc, 1984. 684 pp. ISBN: 978-0-471-87545-1.

[Pic19] Matthieu Picantin. "Automatic Semigroups vs Automaton Semigroups." In: 46th International Colloquium on Automata, Languages, and Programming (ICALP 2019). (Patras, Greece, July 8–12, 2019). Ed. by Christel Baier, Ioannis Chatzigiannakis, Paola Flocchini, and Stefano Leonardi. Vol. 132. Leibniz International Proceedings in Informatics (LIPIcs). Dagstuhl, Germany: Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2019, 124:1–124:15. ISBN: 978-3-95977-109-2. DOI: 10.4230/LIPIcs.ICALP.2019.124.

[Rob71] Raphael M. Robinson. "Undecidability and nonperiodicity for tilings of the plane." *Inventiones mathematicae* 12.3 (1971), pp. 177–209. ISSN: 1432-1297. DOI: 10.1007/BF01418780.

[SV11] Dmytro Savchuk and Yaroslav Vorobets. "Automata generating free products of groups of order 2." *Journal of Algebra* 336.1 (2011), pp. 53–66. ISSN: 0021-8693. DOI: 10.1016/j.jalgebra.2011.02.049. arXiv: 0806.4801 [math.GR].

[Sid00] Said N. Sidki. "Automorphisms of one-rooted trees: growth, circuit structure, and acyclicity." *Journal of Mathematical Sciences* 100.1 (2000), pp. 1925–1943. DOI: 10.1007/BF02677504.

[Ste15] Benjamin Steinberg. "On some algorithmic properties of finite state automorphisms of rooted trees." In: *Algorithmic Problems of Group Theory, Their Complexity, and Applications to Cryptography*. AMS Special Session on Algorithmic Problems of Group Theory and Their Complexity and AMS Special Session on Algorithmic Problems of Group Theory and Applications to Information Security. Ed. by Delaram Kahrobaei and Vladimir Shpilrain. Vol. 633. Contemporary Mathematics. American Mathematical Society, 2015, pp. 115–123. ISBN: 978-0-8218-9859-8. DOI: 10.1090/conm/633/12655.

[ŠV12] Zoran Šunić and Enric Ventura. "The conjugacy problem in automaton groups is not solvable." *Journal of Algebra* 364 (2012), pp. 148–154. ISSN: 0021-8693. DOI: 10.1016/j.jalgebra.2012.04.014.